



UNIVERSITÀ
DI TRENTO

Department of
Information Engineering and Computer Science

Doctoral School in
Information and Communication Technology

RIEMANNIAN DEEP LEARNING: MODULES, NETWORKS, AND GEOMETRIES

Ziheng Chen

Advisor

Prof. Nicu Sebe
Università di Trento

ELLIS Co-supervisor

Prof. Bernhard Schölkopf
Max Planck Institute for Intelligent Systems

September 2026

Publications

(† *corresponding author*, ‡ *equal contribution*, ♣ *equal supervision*)

The thesis is based on the following publications:

- Chapter 3:
 - [1] **Ziheng Chen**, Yue Song, Yunmei Liu, and Nicu Sebe. “**A Lie Group Approach to Riemannian Batch Normalization.**” ICLR 2024.
 - [2] **Ziheng Chen**, Yue Song, Xiao-Jun Wu, and Nicu Sebe. “**Gyrogroupp Batch Normalization.**” ICLR 2025.
- Chapter 4:
 - [3] **Ziheng Chen**, Yue Song, Gaowen Liu, Ramana Rao Kompella, Xiao-Jun Wu, and Nicu Sebe. “**Riemannian Multinomial Logistics Regression for SPD Neural Networks.**” CVPR 2024.
 - [4] **Ziheng Chen**, Yue Song, Rui Wang, Xiao-Jun Wu, and Nicu Sebe. “**RMLR: Extending Multinomial Logistic Regression into General Geometries.**” NeurIPS 2024.
- Chapter 5:
 - [5] **Ziheng Chen**[†], Zihan Su[‡], Bernhard Schölkopf, and Nicu Sebe. “**Proper Velocity Neural Networks.**” ICLR 2026.
 - [6] **Ziheng Chen**, Bernhard Schölkopf, and Nicu Sebe. “**Hyperbolic Busemann Neural Networks.**” CVPR 2026.
 - [7] **Ziheng Chen**, Xiao-Jun Wu, Bernhard Schölkopf, and Nicu Sebe. “**Riemannian Networks over Full-Rank Correlation Matrices.**” ICML 2026.
- Chapter 6:
 - [8] **Ziheng Chen**, Yue Song, Tianyang Xu, Zhiwu Huang, Xiao-Jun Wu, and Nicu Sebe. “**Adaptive Log-Euclidean Metrics for SPD Matrix Learning.**” IEEE TIP 2024.
 - [9] **Ziheng Chen**, Yue Song, Xiao-Jun Wu, and Nicu Sebe. “**Fast and Stable Riemannian Metrics on SPD Manifolds via Cholesky Product Geometry.**” ICLR 2026.

The following papers are published but are not included in this thesis:

- (10) **Ziheng Chen**, Yue Song, Xiao-Jun Wu, Gaowen Liu, and Nicu Sebe. “**Understanding Matrix Function Normalizations in Covariance Pooling through the Lens of Riemannian Geometry.**” ICLR 2025.
- (11) Shaocheng Jin, Tao Zhou, Rui Wang, **Ziheng Chen**, Xiaoqing Luo, Xiao-Jun Wu, and Josef Kittler. “Towards Robust EEG Decoding Based on Riemannian Self-Attention.” KDD 2026.
- (12) Rui Wang, Zihao Bi, Chen Hu, Xiaoning Song, Xiao-Jun Wu, Nicu Sebe, and **Ziheng Chen**[†]. “Riemannian Graph Convolutional Network for Skeleton-Based Two-Person Interaction Recognition.” IJCAI 2026.

-
- (13) Xianglong Shi[‡], **Ziheng Chen**^{†,‡}, Yunhan Jiang, and Nicu Sebe. “**Intrinsic Lorentz Neural Network.**” ICLR 2026.
 - (14) Shanglin Li, Shiwen Chu, Okan Koç, Yi Ding, Qibin Zhao, Motoaki Kawanabe, and **Ziheng Chen**[†]. “**HEEGNet: Hyperbolic Embeddings for EEG.**” ICLR 2026.
 - (15) Chen Hu[‡], **Ziheng Chen**[‡], Rui Wang, Yefeng Zheng, and Nicu Sebe. “**Riemannian High-Order Pooling for Brain Foundation Models.**” ICLR 2026.
 - (16) Rui Wang, Yuting Jiang, Xiaoqing Luo, Xiao-Jun Wu, Nicu Sebe, and **Ziheng Chen**[†]. “**Wasserstein-Aligned Hyperbolic Multi-View Clustering.**” AAAI 2026 (Oral).
 - (17) Rui Wang, Chen Hu, Xiaoning Song, Xiao-Jun Wu, Nicu Sebe, and **Ziheng Chen**[†]. “**Towards a General Attention Framework on Gyrovector Spaces for Matrix Manifolds.**” NeurIPS 2025.
 - (18) Rui Wang, Shaocheng Jin, Zhenyu Cai, **Ziheng Chen**[†], Xiao-Jun Wu[†], and Josef Kittler. “**Learning a Better SPD Network for Signal Classification: A Riemannian Batch Normalization Method.**” IEEE TNNLS 2025.
 - (19) Chen Hu, Rui Wang[♣], Xiaoning Song, Tao Zhou, Xiao-Jun Wu, Nicu Sebe, and **Ziheng Chen**[♣]. “**A Correlation Manifold Self-Attention Network for EEG Decoding.**” IJCAI 2025.
 - (20) Rui Wang, Shaocheng Jin, **Ziheng Chen**[†], Xiaoqing Luo, and Xiao-Jun Wu. “**Learning to Normalize on the SPD Manifold under Bures-Wasserstein Geometry.**” CVPR 2025.
 - (21) Rui Wang, Jiayao Jin, **Ziheng Chen**[†], Cong Wu[†], Xiao-Jun Wu, and Nicu Sebe. “**Structural Topology Refinement Network for Skeleton-Based Action Recognition.**” IEEE TIM 2025.
 - (22) Rui Wang, Chen Hu, **Ziheng Chen**[†], Xiao-Jun Wu[†], and Xiaoning Song. “**A Grassmannian Manifold Self-Attention Network for Signal Classification.**” IJCAI 2024.
 - (23) Rui Wang, Xiao-Jun Wu, **Ziheng Chen**, Cong Hu, and Josef Kittler. “**SPD Manifold Deep Metric Learning for Image Set Classification.**” IEEE TNNLS 2024.

Abstract

Recently, deep neural networks operating on manifold-valued representations have garnered significant attention across various machine learning applications. However, many basic neural components remain tied to particular manifolds or rely on Euclidean approximations, while the underlying geometry can make repeated computations costly or numerically unstable. This thesis addresses these limitations by developing reusable modules, exploiting manifold-specific structures when general constructions are insufficient, and introducing fast and stable geometries. We first generalize batch normalization and classification beyond individual manifolds. For normalization, we develop a framework on Lie groups, with theoretical control over Riemannian sample means and variances. To extend this principle beyond Lie groups, we introduce pseudo-reductive gyrogroups, which generalize classical gyrogroups and Lie groups, and build a normalization framework on this structure for a wider range of manifolds. For classification, we first extend Euclidean Multinomial Logistic Regression (MLR), which consists of a fully connected layer followed by softmax, to Symmetric Positive Definite (SPD) manifolds with flat metrics. We then use Riemannian trigonometry to extend MLR to general Riemannian manifolds. When a general formulation cannot exploit useful structure, we design networks for particular representations. For stable hyperbolic deep learning, we use Proper Velocity, an unconstrained representation of hyperbolic space, and develop its geometry and core neural layers. We also use Busemann functions to build intrinsic and efficient hyperbolic classification and fully connected layers. For full-rank correlation matrices, a normalized alternative to SPD matrices, we construct networks that operate directly on the manifold and derive accurate gradients for end-to-end training under two correlation geometries. Finally, we study how the geometry itself can improve learning on SPD manifolds. To move beyond fixed metrics, we make the metric learnable through parameterized matrix logarithms, allowing it to adapt to data and network dynamics with little additional computation. To improve efficiency and numerical stability, we exploit the product structure of Cholesky factors to construct SPD metrics with fast and stable closed-form operators. The proposed methods are supported by theoretical analysis and validated through numerical experiments and empirical applications in vision, signal processing, graph learning, and genomics.

Keywords

Riemannian deep learning, geometric deep learning, Riemannian manifolds, matrix manifolds, Lie groups, constant-curvature manifolds

Acknowledgements

My Ph.D. journey would not have been possible without the guidance, collaboration, and encouragement of many people. First and foremost, I would like to express my deepest gratitude to my advisor, Prof. Nicu Sebe. He gave me the freedom to pursue the questions that genuinely interested me and created an open research environment in which I could explore new directions with confidence. At the same time, he was always generous with his time and consistently offered encouragement and guidance in our discussions. His support extended well beyond research. He advised me on career decisions, helped me engage with the research community, and supported my scholarship applications. This balance between intellectual freedom and dependable support has profoundly shaped both the work presented in this thesis and the researcher I have become.

I am also deeply grateful to Prof. Bernhard Schölkopf. During my research stay with him, his generosity and intellectual openness allowed me to explore distributional geometry in depth and to develop new perspectives on its role in machine learning. I value his insight, trust, and support. I have greatly enjoyed discussing research questions with him and feel fortunate that our collaboration will continue during my postdoctoral research.

I would also like to thank Prof. Xiaojun Wu, who was my master's advisor. We continued to collaborate throughout my Ph.D., and I deeply appreciate the generous help, guidance, and encouragement he provided along the way.

My sincere thanks also go to my collaborators Yue Song, Rui Wang, and Shanglin Li, as well as to the students I mentored: Zihan Su, Xianglong Shi, Youxing Li, Ying Zhang, Chen Hu, Shaocheng Jin, Zihao Bi, and Yuting Jiang. Our discussions and joint efforts have been an essential part of my Ph.D. experience. I have learned a great deal from each of them, and I am grateful for the ideas, dedication, and enthusiasm they brought to our work together.

Finally, I would like to thank my family. My deepest and most personal thanks go to my girlfriend, Yunmei Liu. Throughout this journey, she has supported me with patience and warmth and has always believed in me. She has made the difficult moments easier and the joyful moments more meaningful. She has also brought more good fortune into my life than I could ever have imagined. I am profoundly grateful to have her by my side.

Contents

Notation and Conventions	xxi
1 Introduction	1
1.1 Riemannian Deep Learning	1
1.2 Contributions and Outlines	3
1.3 Summary of Papers Excluded from the Thesis	5
2 Mathematical Background	9
2.1 Introduction	9
2.2 Topology	9
2.3 Differential Geometry	12
2.4 Riemannian Geometry	17
2.5 Metric Geometry	25
2.6 Algebraic Structures on Manifolds	29
2.7 Riemannian Optimization	34
2.8 Matrix Functions	35
2.8.1 Matrix Functions and Differentials	35
2.8.2 Backpropagation Through Matrix Functions	36
2.9 Example Manifolds	37
2.9.1 Symmetric Positive Definite Manifolds	37
2.9.2 Full-Rank Correlation Manifolds	39
2.9.3 Grassmannian Manifolds	48
2.9.4 Special Orthogonal Groups	50
2.9.5 Constant-Curvature Manifolds	50
3 Riemannian Batch Normalization	55
3.1 Introduction	55
3.2 Lie Group Batch Normalization	56
3.2.1 Introduction	56
3.2.2 Preliminaries	58
3.2.3 Revisiting Normalization	59
3.2.4 LieBN	61
3.2.5 Manifestations	65
3.2.6 Experiments	71
3.3 Gyrogroup Batch Normalization	76

3.3.1	Introduction	76
3.3.2	Pseudo-Reductive Gyrogroups	79
3.3.3	GyroBN on Pseudo-Reductive Gyrogroups	84
3.3.4	Instantiations	86
3.3.5	Experiments	99
3.4	Conclusion	107
4	Riemannian Multinomial Logistic Regression	109
4.1	Introduction	109
4.2	Multinomial Logistic Regression on SPD Manifolds	110
4.2.1	Introduction	110
4.2.2	SPD Multinomial Logistic Regression	111
4.2.3	SPD MLRs under Deformed LEM and LCM	115
4.2.4	Rethinking the Existing LogEig Classifier	116
4.3	Extension to General Riemannian Manifolds	117
4.3.1	Introduction	117
4.3.2	Riemannian Multinomial Logistic Regression	119
4.3.3	SPD Multinomial Logistic Regressions	123
4.3.4	Lie Multinomial Logistic Regression	126
4.4	Experiments	127
4.4.1	Experiments on the Proposed SPD MLRs	128
4.4.2	Experiments on the Proposed Lie MLR	131
4.5	Conclusion	132
5	Riemannian Neural Networks	133
5.1	Introduction	133
5.2	Proper Velocity Neural Networks	134
5.2.1	Introduction	134
5.2.2	Preliminaries	135
5.2.3	Proper Velocity Geometry	136
5.2.4	Proper Velocity Neural Networks	139
5.2.5	Connections to the Hyperboloid	142
5.2.6	Experiments	144
5.3	Hyperbolic Busemann Neural Networks	149
5.3.1	Introduction	149
5.3.2	Preliminaries	151
5.3.3	Busemann Multinomial Logistic Regression	152
5.3.4	Busemann Fully Connected Layer	156
5.3.5	Experiments	159
5.4	Full-Rank Correlation Networks	163
5.4.1	Introduction	163
5.4.2	Log-Euclidean Correlation Layers	165
5.4.3	Poly-Hyperbolic-Cholesky Layers	169
5.4.4	Backpropagation over Correlation Geometries	171

5.4.5	Experiments	172
5.5	Conclusion	183
6	Fast and Stable Geometries on SPD Manifolds	185
6.1	Introduction	185
6.2	Adaptive Log-Euclidean Metrics	186
6.2.1	Introduction	186
6.2.2	Adaptive Log-Euclidean Metrics	187
6.2.3	Parameter Learning	195
6.2.4	Experiments	197
6.3	Product Cholesky Metrics	204
6.3.1	Introduction	204
6.3.2	Preliminaries	205
6.3.3	Product Geometries on the Cholesky	206
6.3.4	Geometries on the SPD Manifold	211
6.3.5	Applications to SPD Neural Networks	213
6.3.6	Experiments	213
6.4	Conclusion	220
7	Conclusion and Future Work	221
7.1	Conclusion	221
7.2	Future Work	222
	Bibliography	223
A	Experimental Details and Additional Discussions	245
A.1	Data Sets	245
A.1.1	Skeleton-Based Action Recognition and Gesture Data Sets	245
A.1.2	Radar and EEG Signal Data Sets	246
A.1.3	Image Classification Data Sets	246
A.1.4	Graph Data Sets	246
A.1.5	Genomic Sequence Data Sets	247
A.2	Backbone Networks	248
A.2.1	Backbone Networks on the SPD Manifold	248
A.2.2	Backbone Networks on the Grassmannian	249
A.2.3	Backbone Networks on Rotation Matrices	250
A.2.4	Backbone Networks on Hyperbolic Spaces	250
A.2.5	Riemannian Residual Network Backbones	252
A.3	Experimental Details	253
A.3.1	Lie Group Batch Normalization	253
A.3.2	Gyrogroup Batch Normalization	257
A.3.3	Riemannian Multinomial Logistic Regression	258
A.3.4	Proper Velocity Neural Networks	261
A.3.5	Hyperbolic Busemann Neural Networks	263
A.3.6	Full-Rank Correlation Networks	266

A.3.7	Adaptive Log-Euclidean Metrics	269
A.3.8	Product Cholesky Metrics	271
A.4	Additional Discussions	273
A.4.1	Riemannian Multinomial Logistic Regression	273
A.4.2	Hyperbolic Busemann Neural Networks	282
A.4.3	Full-Rank Correlation Networks	286
A.4.4	Adaptive Log-Euclidean Metrics	289
A.4.5	Product Cholesky Metrics	290
B	Proofs	291
B.1	Mathematical Background	291
B.1.1	Proof of Thm. 57	291
B.2	Lie Group Batch Normalization	291
B.2.1	Proof of Thm. 61	291
B.2.2	Proof of Thm. 62	292
B.2.3	Proof of Thm. 65	292
B.2.4	Proof of Thm. 66	293
B.2.5	Proof of Thm. 67	293
B.2.6	Proof of Thm. 68	294
B.2.7	Proof of Thm. 69	294
B.2.8	Proof of Thm. 70	296
B.2.9	Proof of Thm. 72	297
B.3	Gyrogroup Batch Normalization	298
B.3.1	Proof of Thm. 74	298
B.3.2	Proof of Thm. 75	299
B.3.3	Proof of Thm. 77	300
B.3.4	Proof of Thm. 78	300
B.3.5	Proof of Thm. 79	302
B.3.6	Proof of Thm. 80	303
B.3.7	Proof of Thm. 81	304
B.3.8	Proof of Thm. 83	305
B.3.9	Proof of Thm. 86	306
B.3.10	Proof of Thm. 88	309
B.3.11	Proof of Thm. 89	309
B.3.12	Proof of Thm. 90	312
B.3.13	Proof of Thm. 91	312
B.3.14	Proof of Thm. 92	314
B.3.15	Proof of Thm. 93	318
B.3.16	Proof of Thm. 95	318
B.3.17	Proof of Thm. 97	318
B.3.18	Proof of Thm. 98	321
B.3.19	Proof of Thm. 99	322
B.4	SPD Multinomial Logistic Regression	326
B.4.1	Proof of Thm. 104	326

B.4.2	Proof of Thm. 105	327
B.4.3	Proof of Thm. 106	327
B.4.4	Proof of Thm. 107	327
B.4.5	Proof of Thm. 108	328
B.4.6	Proof of Thm. 109	328
B.4.7	Proof of Thm. 111	329
B.5	Riemannian Multinomial Logistic Regression	332
B.5.1	Proof of Thm. 113	332
B.5.2	Proof of Thm. 114	333
B.5.3	Proof of Thm. 116	333
B.5.4	Proof of Thm. 117	333
B.5.5	Proof of Thm. 119	338
B.5.6	Proof of Thm. 120	338
B.6	Proper Velocity Neural Networks	338
B.6.1	Derivation of the Proper Velocity Metric	338
B.6.2	Proof of Thm. 121	339
B.6.3	Proof of Thm. 122	341
B.6.4	Proof of Thm. 123	342
B.6.5	Proof of Thm. 124	350
B.6.6	Proof of Thm. 125	350
B.6.7	Proof of Thm. 126	354
B.6.8	Proof of Thm. 127	359
B.6.9	Proof of Thm. 128	359
B.6.10	Proof of Thm. 129	360
B.7	Hyperbolic Busemann Neural Networks	363
B.7.1	Proof of Thm. 130	363
B.7.2	Proof of Thm. 132	364
B.7.3	Proof of Thm. 135	366
B.7.4	Proof of Thm. 136	367
B.7.5	Proof of Thm. 137	368
B.8	Full-Rank Correlation Networks	370
B.8.1	Proof of Thm. 138	370
B.8.2	Proof of Thm. 139	372
B.8.3	Proof of Thm. 142	373
B.8.4	Proof of Thm. 143	376
B.8.5	Proof of Thm. 145	377
B.8.6	Proof of Thm. 146	379
B.8.7	Proof of Thm. 144	380
B.9	Adaptive Log-Euclidean Metrics	380
B.9.1	Proof of Thm. 147	380
B.9.2	Proof of Thm. 148	381
B.9.3	Proof of Thm. 149	381
B.9.4	Proof of Thm. 150	381
B.9.5	Proof of Thm. 152	382

B.9.6	Proof of Thm. 154	382
B.9.7	Proof of Thm. 155	383
B.9.8	Proof of Thm. 156	384
B.9.9	Proof of Thm. 157	384
B.9.10	Proof of Thm. 158	385
B.9.11	Proof of Thm. 159	385
B.9.12	Proof of Thm. 160	385
B.9.13	Proof of Thm. 161	385
B.9.14	Proof of Thm. 163	386
B.9.15	Proof of Thm. 164	386
B.9.16	Proof of Thm. 167	386
B.9.17	Proof of Thm. 165	388
B.9.18	Proof of Thm. 166	389
B.10	Product Cholesky Metrics	390
B.10.1	Proof of Thm. 169	390
B.10.2	Proof of Thm. 170	391
B.10.3	Proof of Thm. 172	393
B.10.4	Proof of Thm. 173	393
B.10.5	Proof of Thm. 174	395
B.10.6	Proof of Thm. 177	396
B.10.7	Proof of Thm. 179	396

List of Tables

2.1	Euclidean prototypes for topological concepts.	12
2.2	Euclidean prototypes for differential-geometric concepts.	18
2.3	Euclidean prototypes for Riemannian-geometric concepts.	24
2.4	Euclidean prototypes for metric-geometric concepts.	29
2.5	Lie group structures and associated Riemannian operators on $\mathcal{S}_{++}^n$. . .	39
2.6	Riemannian operators of (θ, α, β) -EM and BWM on $\mathcal{S}_{++}^n$	39
2.7	Isometric prototype spaces and diffeomorphisms on the correlation manifold.	46
2.8	Vector operations and Riemannian operators under ECM and LECM. .	46
2.9	Vector operations and Riemannian operators under OLM and LSM. . .	46
2.10	Riemannian operators on the Grassmannian under ONB and PP. . . .	49
2.11	Gyro operators on the Grassmannian under ONB and PP.	49
2.12	Lie group structures and Riemannian operators on rotation matrices. .	50
2.13	Gyro operators on stereographic and Beltrami–Klein models.	52
2.14	Riemannian operator templates for stereographic and radius constant-curvature coordinates.	54
3.1	Review of SPD Lie groups and invariant metrics.	59
3.2	Review of the rotation Lie group and invariant metric.	59
3.3	Review of full-rank correlation Lie groups and invariant metrics. . . .	60
3.4	Summary of some representative RBN methods.	60
3.5	Summary of LieBN types.	65
3.6	Key operators in calculating LieBN on SPD manifolds.	68
3.7	Key operators in calculating LieBN on the rotation matrices.	68
3.8	Summary of LieBN on the correlation.	70
3.9	10-fold average results of SPDNet with and without SPDBN or LieBN. .	72
3.10	Cross-validation results of TSMNet with SPDDSMNB and DSMLieBN. .	73
3.11	Results of LieNet with or without rotation LieBN.	75
3.12	Results of SPDNet with or without correlation LieBN under different invariant metrics.	76
3.13	Comparison of previous RBN methods with GyroBN, where $\mathbf{M}$ and $\mathbf{V}$ denote the sample mean and variance.	77
3.14	Summary of operators for GyroBN across representative manifolds. . .	99

3.15	Efficiency (in μs) of gyroaddition on the radius manifold: closed form versus Riemannian definition. Values in parentheses indicate the runtime of the closed-form implementation as a percentage of the corresponding Riemannian implementation. The best results are bold .	100
3.16	Comparison of GyroBN against other Grassmannian BN methods under the GyroGr backbone. Here, accuracy is reported as a percentage, fit time denotes the average training time per epoch (s/epoch), and #Params is reported in millions. Values in parentheses specify the dimension of the Grassmannian input to the BN layer. The largest number of parameters is marked in red.	103
3.17	Comparison of GyroBN against LRBN across five CCSs. ROC is the testing AUC reported as a percentage, fit time is measured in s/epoch, and #Params is reported in millions. When LRBN degenerates the backbone network, the results are highlighted with red.	106
3.18	Comparison of CorNet with or without RBN layers. Accuracy is reported as a percentage, fit time is measured in s/epoch, and #Params is reported in millions.	107
4.1	Several MLRs on different geometries are special cases of our MLR.	123
4.2	Properties of deformed metrics on SPD manifolds ($\theta \neq 0$ and $\min(\alpha, \alpha + n\beta) > 0$).	124
4.3	Comparison of SPDNet with LogEig against SPD MLRs on the Radar data set. The best results are bold .	127
4.4	Comparison of SPDNet with LogEig against SPD MLRs on the HDM05 data set.	127
4.5	Inter-session TSMNet results on Hinss2021.	127
4.6	Inter-subject TSMNet results on Hinss2021.	128
4.7	Comparison of LogEig against SPD MLRs under the RResNet architecture.	129
4.8	Comparison of LogEig against SPD MLRs under the SPDGCN architecture.	130
4.9	Comparison of LogEig against SPD MLRs for direct classification.	130
4.10	Results of LogEig MLR against Lie MLR under the LieNet architecture.	131
5.1	Failure and violation rates (%) of $r \otimes_{\mathcal{H}} x$ in FP32.	144
5.2	$\ \text{Log}_0(\text{Exp}_0(v)) - v\ $.	145
5.3	Gradient magnitude $\ \nabla_x f_r(x)\ $ across varying radii.	145
5.4	Top-1 image classification accuracy (%) of hyperbolic MLRs on ResNet-18. The best results are bold . δ represents the δ -hyperbolicity (lower is more hyperbolic), which comes from Bdeir et al. [15, Tab. 1].	146
5.5	Accuracies of hyperbolic networks on graph learning. The best results are bold . δ represents the δ -hyperbolicity (lower is more hyperbolic).	146
5.6	Results of Tangent FC (TFC) vs PV FC, and Tangent BN (TBN) vs GyroBN.	147
5.7	Comparison of methods in calculating mean and variance in PV GyroBN. Time is measured in milliseconds per training epoch.	147

5.8	Ablations on PVNN with or without exponential map for the input PV feature.	148
5.9	Ablations on PV activations.	148
5.10	Comparison in MCC of hyperbolic and Euclidean convolutional networks, including PVCNN, on TEB data sets.	149
5.11	Comparison of C -class MLR. In Dist, Real means the point-to-hyperplane distance is the real distance, obtained by $\inf_{y \in H} d(x, y)$, where H is a hyperplane and d is the geodesic distance; Pseudo denotes a surrogate that coincides with the real distance only in Euclidean geometry. Compact params indicate whether each logit avoids an additional manifold-valued parameter. Batch efficiency indicates whether the MLR can avoid inefficient per-class loops in implementation (see Sec. A.4.2.1). In #Params, we highlight the heaviest in red . In FLOPs, we mark the slowest in red and the fastest in green	155
5.12	Comparison of hyperbolic FC layers. For simplicity, BFC layers do not involve the gyroaddition and assume ϕ is the identity map, which is in line with the Möbius and Lorentz FC layers.	158
5.13	Top-1 image classification accuracy (%) of MLR methods on the ResNet-18 backbone. The best results within each hyperbolic model are bold . The slowest MLR and largest parameter count are shown in red	159
5.14	Genomic MCC of MLR methods under the CNN backbone. The best results within each hyperbolic model are bold	160
5.15	Fit time (s/epoch) on genome sequence learning. The fastest times are bold and the slowest ones are red	161
5.16	Comparison of hyperbolic FC layers on link prediction. The best results within each hyperbolic model are bold	161
5.17	Node classification F1 scores of hyperbolic MLRs on the HGCN backbone, where δ denotes graph hyperbolicity (lower is more hyperbolic). The best results within each hyperbolic model are bold	162
5.18	Efficiency comparison: fit time (s/epoch) and parameter count. Slowest results and largest parameter counts are in red	163
5.19	Correspondence between Euclidean and correlation-based layers. For convolution, kernel-based FC refers to applying a convolution kernel to a receptive field, which is an FC transformation.	165
5.20	Five-fold results and training time per epoch on four data sets. The top 3 results are highlighted with red , blue , and cyan . * denotes reproduced results due to missing official code.	173
5.21	Ablations on mixed geometries. Each row shows the metric used for Convolution (Conv), and each column is the metric for MLR. The diagonal entries indicate configurations where both layers use the same metric. The best result in each row is bold	174
5.22	SPDNet: SPD vs. correlation.	175

5.23	Comparison of SPDMLR-Trivlz on raw covariances against CorMLR on raw correlations on all three data sets. The input matrix dimensions are 93×93 , 63×63 , and 20×20 , respectively.	175
5.24	SPD networks with or without normalized SPD inputs.	180
5.25	Comparison of CorNet with or without activations.	181
5.26	Average runtime (s) of a single forward pass in CorNet under different metrics and input dimensions. The best results are bold	182
6.1	Parameter learning for the general matrix logarithm and exponential.	196
6.2	Results of ALog on the HDM05 data set. The best results are bold	198
6.3	Results of ALog on the FPHA data set.	199
6.4	Results of ALog on the AFEW data set.	200
6.5	Results of fixed bases on the HDM05 and FPHA data sets.	201
6.6	Comparison of RBN methods on the HDM05 data set.	202
6.7	Experiments on RResNet under different geometries.	203
6.8	Comparison of Gyro MLRs on the NTU60 data set.	203
6.9	Riemannian and gyro operators of different metrics on the Cholesky manifold. For the diagonal log metric, $\log(\cdot)$ and $\exp(\cdot)$ are diagonal logarithm and exponentiation.	211
6.10	SPD MLRs under different metrics on the SPDNet backbone. The best results are bold	214
6.11	SPD MLRs on the GyroSPD backbone.	214
6.12	Results on residual blocks.	215
6.13	Failure probabilities (%) of geodesics under different metrics with small eigenvalues in $L \in \mathcal{L}_{++}^n$. An output matrix containing any INF or NAN is considered a failure. Here, DLM denotes the diagonal log metric, while DPM and DBWM denote θ -DPM and θ -DBWM, respectively.	216
6.14	Swelling effects of geodesic SPD interpolations. Deeper greens indicate greater swelling.	218
6.15	Number of matrix functions required per sample for a C -class SPD MLR. Spectral matrix functions include matrix logarithm, matrix power, and the Lyapunov operator.	218
6.16	Asymptotic per-sample complexity of a C -class SPD MLR for an $n \times n$ input SPD matrix.	219
6.17	Average runtime (in seconds) of one SPD MLR training step across different matrix dimensions.	220
A.1	Summary statistics for the graph data sets.	246
A.2	Summary statistics for TEB.	247
A.3	Summary statistics for the adopted GUE data sets.	248
A.4	Matrix powers in LieBN-Cor under different metrics on each data set.	257
A.5	(θ, α, β) of SPD MLRs on the SPDGCN backbone.	259
A.6	Candidate values for hyperparameters in SPD MLRs.	260
A.7	Hyperparameters for PVNN that vary across graph data sets.	262
A.8	Hyperparameters for PVNN that are shared across graph data sets.	262

A.9 Summary of the hyperbolic layers used in the graph node classification models.	263
A.10 Hyperparameters for TEB.	263
A.11 Summary of hyperparameters used in the image classification task. . . .	264
A.12 Hyperparameters for genome sequence learning.	265
A.13 Hyperparameters for node classification on Disease, Airport, PubMed, and Cora.	265
A.14 Hyperparameters in CorNets.	267
A.15 Hyperparameter θ in θ -PCM and θ -BWCM. It is selected from the candidate values used for SPDNet and GyroSPD.	272
A.17 Comparison of point-to-hyperplane distances. Real means the point-to-hyperplane distance is the real distance, obtained by $\inf_{y \in H} d(x, y)$ with H as a hyperplane and d as the geodesic distance. Instead, Pseudo means the point-to-hyperplane distance is a surrogate, which only equals the real distance in Euclidean geometry.	282
A.16 Comparison of hyperplanes. Compact params indicate whether the parameterization requires an additional manifold-valued point.	282

List of Figures

2.1	The black stars denote 2×2 correlation matrices, while the red, green, and blue dots denote corresponding SPD matrices. The black dots denote the boundary of the SPD cone.	40
3.1	Illustration of LieBN on the SPD, rotation, and correlation Lie groups.	56
3.2	Minimal examples of applying LieBN.	58
3.3	Visualization of input and output SPD matrices in LieBN.	74
3.4	Test accuracy curves of LieNet with rotation LieBN.	75
3.5	Illustration of GyroBN on manifold-valued data. Blue points, green points, and the red dashed curves indicate the input samples, normalized outputs, and data distributions, respectively.	77
3.6	Minimal examples of applying GyroBN.	79
3.7	Comparison of derivation logic for gyrotranslation isometries.	80
3.8	Visualization of GyroBN across different geometries. Blue and green points represent input and normalized data, respectively. Red and cyan points denote the input and output batch means. Black points mark the manifold boundary, and the gray surface depicts the manifold.	101
3.9	Training and testing curves of 1-block GyroGr on two NTU data sets.	104
4.1	Conceptual illustration of SPD hyperplanes induced by (α, β) -LEM and θ -LCM. In each subfigure, the black dots are SPSD matrices, denoting the boundary of $\mathcal{S}_{++}^2$, while the blue, red, and yellow dots denote three SPD hyperplanes.	117
4.2	Illustration of the deformation (left) and Venn diagram (right) of metrics on SPD manifolds, where IEM, SREM, and $\frac{1}{4}$ PAM denote Inverse Euclidean Metric, Square Root Euclidean Metric, and Polar Affine Metric scaled by $1/4$, respectively.	123
4.3	Conceptual illustration of SPD hyperplanes induced by five families of Riemannian metrics. The black dots denote the boundary of $\mathcal{S}_{++}^2$	124
4.4	Conceptual illustration of a Lie hyperplane. Each pair of antipodal black dots corresponds to a rotation matrix with an Euler angle of π , while the green dots denote a Lie hyperplane.	127
5.1	Illustration: red curves are different horospheres of B^v	152
5.2	Validation accuracy curves on ImageNet-1k.	159

5.3	Illustration of the Log-Euclidean 1D convolution with two kernels. The 3-channel input is first split into two receptive fields along the channel dimension. In each receptive field, two kernels are applied to the product space.	168
5.4	Illustration of the PHCM convolution and MLR. The multi-channel input correlation matrices are denoted as $\{C^i\}_{i=1}^c$. For the convolutional layer, the illustration focuses on the transformation within a receptive field and assumes a single-channel output.	170
5.5	Illustration of the decision hyperplanes in the correlation MLRs under five different geometries. The 3×3 correlation manifold can be embedded as an open elliptope in $\mathbb{R}^3$, by visualizing the strictly lower triangular part of each $C \in \text{Cor}^+(3)$. The black dots denote the boundary. The PHCM hyperplane is defined by the one in the β -concatenated Poincaré space.	174
5.6	Distribution of per-sample coefficients of variation of diagonal variances on FPHA. Higher values indicate stronger diagonal variability, which could cause nuisance noise.	176
5.7	Distribution of per-sample coefficients of variation of diagonal variances on HDM05. Higher values indicate stronger diagonal variability, which could cause nuisance noise.	177
5.8	Distribution of ratios of diagonal to off-diagonal entries on FPHA.	178
5.9	Distribution of ratios of diagonal to off-diagonal entries on HDM05.	178
6.1	Accuracy curves on the FPHA data set.	199
6.2	Visualization of parameters in the ALog layer on the HDM05 and FPHA data sets.	201
6.3	Geodesic interpolation of SPD matrices under different Riemannian metrics. Each 3×3 SPD matrix can be visualized as an ellipsoid [9]. The two endpoints are fixed across all metrics.	217
B.1	Illustration of the Euclidean spaces $\text{LT}^0(m)$, $\text{Hol}(m)$ and $\text{Row}_0(m)$, where $\star$ can be obtained by symmetry.	375

Notation and Conventions

This chapter collects the notation used throughout the thesis.

General Sets and Linear Algebra

$\mathbb{R}, \mathbb{R}^n$	The real numbers and the n -dimensional Euclidean space.
$\mathbb{R}_+, \mathbb{R}_{++}$	The non-negative and positive real numbers.
$I_n, \mathbf{0}_n, \mathbf{1}_n, \mathbf{1}, \mathbf{0}_{n \times n}$	The $n \times n$ identity matrix, the n -dimensional zero vector, the n -dimensional all-one vector (also written $\mathbf{1}$ when its dimension is clear), and the $n \times n$ zero matrix. Dimension subscripts may be omitted when they are clear from context.
$\langle x, y \rangle, \ x\ $	The Euclidean inner product and the induced norm.
$\ A\ , \text{tr}(A), \text{rank}(A)$	The Frobenius norm, trace, and rank of a matrix.
$\text{diag}(x), \text{diag}(A)$	The diagonal matrix induced by a vector and the diagonal part of a matrix, with the intended meaning specified by context.
$\text{argmin}, \text{argmax}$	The minimizer and maximizer operators.

Manifold Geometry

X, Y	Generic sets or topological spaces.
$\mathcal{X}$	A generic metric space.
$\mathcal{M}, \mathcal{N}$	Smooth or Riemannian manifolds.
$C^\infty(\mathcal{M})$	The space of smooth real-valued functions on $\mathcal{M}$.
$T_p\mathcal{M}$	The tangent space of $\mathcal{M}$ at p .
g_p	The Riemannian metric at p , viewed as an inner product on $T_p\mathcal{M}$.

g^L, g^R	Left- and right-invariant Riemannian metrics on a Lie group.
$\langle v, w \rangle_p, \ v\ _p$	The Riemannian inner product and norm in $T_p\mathcal{M}$.
$d_{\mathcal{M}}(x, y)$	The geodesic distance between $x, y \in \mathcal{M}$.
$\alpha, L(\alpha)$	A general curve on a space and its length.
γ	A geodesic or geodesic ray.
η	The momentum parameter used in LieBN and GyroBN running-statistics updates.
M_K^n, d_K, D_K	The n -dimensional model space of constant curvature K , its distance, and the diameter of M_K^2 .
$\text{CAT}(K)$	A metric space whose geodesic triangles satisfy the $\text{CAT}(K)$ comparison inequality.
π_C	The orthogonal projection onto a convex subset C of a $\text{CAT}(0)$ space.
$\partial\mathcal{X}$	The boundary at infinity of a metric space, defined by equivalence classes of asymptotic geodesic rays when available.
$B^\gamma(x), HB_\tau^\gamma, H_\tau^\gamma$	The Busemann function associated with a geodesic ray γ , its horoball, and its horosphere at level τ .
$\text{Exp}_p, \text{Log}_p$	The Riemannian exponential and logarithmic maps at p .
$\text{PT}_{p \rightarrow q}$	Parallel transport from $T_p\mathcal{M}$ to $T_q\mathcal{M}$.
$\mathcal{T}_{p \rightarrow q}$	A vector transport from $T_p\mathcal{M}$ to $T_q\mathcal{M}$.
$H_{a,p}, \tilde{H}_{\tilde{A},P}$	Euclidean and Riemannian hyperplanes used by multinomial logistic regression.
$P_k, A_k, \tilde{A}_k$	Class-wise Riemannian prototype, fixed tangent-space parameter, and transported tangent parameter in RMLR.
$z_k, r_k, v_k(x)$	The Euclidean direction parameter, scalar offset parameter, and class score used by the PV MLR and PV FC layers.
M, B, v^2, s	The Riemannian mean, biasing parameter, variance, and scaling factor used by Riemannian normalization layers.
$f_{*,p}, d_p f(v)$	The differential of a smooth map f at p , written in Chapter 2 as a linear map and, in later computations, equivalently as applied to a tangent vector v .

$\nabla_w f, \text{grad}_w f$	The Euclidean gradient of a scalar objective at a Euclidean parameter w and the Riemannian gradient at $w \in \mathcal{M}$, respectively.
$\mathfrak{X}(\mathcal{M}), \mathfrak{X}(\alpha)$	Smooth vector fields on $\mathcal{M}$ and smooth vector fields along a curve α .
$[X, Y]$	The Lie bracket of smooth vector fields $X, Y \in \mathfrak{X}(\mathcal{M})$.
$D, \frac{D}{dt}, V'$	The Levi-Civita connection and the induced derivative $V' = \frac{D}{dt}(V)$ along a curve.
$\text{FM}(\{x_i\}),$ $\text{WFM}(\{w_i\}, \{x_i\})$	The Fréchet mean and weighted Fréchet mean.

Algebraic and Gyrovector Operations

G, e, E	A group-like algebraic structure and its identity element.
$\oplus$	A binary operation for groups, Lie groups, and gyrogroups.
L_x, R_x	Left and right translations by x on a Lie group.
L_x	Left gyrotranslation by x , defined by $L_x(y) = x \oplus y$.
$\ominus x$	The inverse of x under an addition-like operation. In gyrogroups, this is called the gyro-inverse.
$\odot$	Scalar multiplication in gyrovector spaces.
$\text{gyr}[x, y]$	The gyration generated by x and y .
$\langle x, y \rangle_{\text{gyr}}$	The gyro inner product.
$\ x\ _{\text{gyr}}$	The gyronorm.
$d_{\text{gyr}}(x, y)$	The gyrodistance.
Bar_η	The binary barycenter operator with weight η , used for running mean updates on gyrogroups.

Matrix Manifolds

$\mathcal{S}^n$	The vector space of $n \times n$ symmetric matrices.
$\mathcal{S}_{++}^n$	The manifold of $n \times n$ symmetric positive definite matrices.
$\mathcal{S}_n^+$	The set of $n \times n$ symmetric positive semidefinite matrices.
$\text{Chol}(P)$	The Cholesky factor of an SPD matrix P .

$\mathcal{L}_{++}^n$	The space of $n \times n$ lower triangular matrices with positive diagonal entries.
$\log(P), \exp(A), \log_\alpha(P), \log_\alpha^{-1}(A)$	The matrix logarithm and exponentiation, followed by the general matrix logarithm used by ALEM and its inverse.
$\oplus^{\text{ALE}}, \odot^{\text{ALE}}$	The ALEM-induced element addition and scalar multiplication on $\mathcal{S}_{++}^n$.
$f(S), f_{*,S}, L_f$	A symmetric matrix function induced by a scalar function f , its differential at S , and the corresponding Loewner matrix.
$A \otimes B$	The Hadamard product of matrices A and B with the same size.
$\log_{*,P}, \text{Chol}_{*,P}, \text{Chol}_{*,L}^{-1}$	The differentials of the matrix logarithm, the Cholesky decomposition, and the inverse Cholesky map at the indicated points.
$P_\theta(P), (P_\theta)_{*,P}$	The matrix power map $P \mapsto P^\theta$ on SPD matrices and its differential at P .
$\phi : \mathcal{S}_{++}^n \rightarrow \mathcal{S}^n, \odot_\phi, g^\phi, d^\phi$	An isometry from SPD matrices to a Euclidean space and its induced abelian group operation, pullback Euclidean metric, and geodesic distance.
$\text{Dlog}(D), \psi_{\text{LC}}(P)$	The diagonal element-wise logarithm and the Log-Cholesky map $\psi_{\text{LC}}(P) = \lfloor L \rfloor + \text{Dlog}(\mathbb{D}(L))$, where $L = \text{Chol}(P)$.
$\mathcal{L}_P[V]$	The Lyapunov operator defined by $\mathcal{L}_P[V]P + P\mathcal{L}_P[V] = V$.
$\langle A, B \rangle^{(\alpha,\beta)}, \ A\ ^{(\alpha,\beta)}$	The two-parameter Euclidean inner product on $\mathcal{S}^n$ and its induced norm.
ST	The admissible parameter set $\{(\alpha, \beta) \in \mathbb{R}^2 \mid \min(\alpha, \alpha + n\beta) > 0\}$.
$\lfloor A \rfloor, \mathbb{D}(A)$	The strictly lower triangular part and the diagonal matrix formed from the diagonal of A .
$A_{1/2}$	The Cholesky half-diagonal operator $A_{1/2} = \lfloor A \rfloor + \frac{1}{2}\mathbb{D}(A)$.
AI	Affine-invariant geometry on SPD matrices.
LE	Log-Euclidean geometry on SPD matrices.
LC	Log-Cholesky geometry on SPD matrices.
PE	Power-Euclidean geometry on SPD matrices.

(α, β) -AIM	The two-parameter Affine-Invariant Metric (AIM) on SPD matrices.
(α, β) -LEM	The two-parameter Log-Euclidean Metric (LEM) on SPD matrices.
(θ, α, β) -AIM	The three-parameter Affine-Invariant Metric (AIM) on SPD matrices.
θ -LCM	The parameterized Log-Cholesky Metric (LCM) on SPD matrices.
(θ, α, β) -EM	The three-parameter Euclidean Metric (EM) on SPD matrices.
(a, b) -ALEM	The Adaptive Log-Euclidean Metric on SPD matrices.
θ -PCM	The deformed Power-Cholesky Metric on SPD matrices.
$(\theta, \mathbb{M})$ -BWCM	The deformed Bures–Wasserstein–Cholesky Metric on SPD matrices.
$\oplus^{\text{AI}}, \odot^{\text{AI}}$	AIM-induced SPD gyroaddition and scalar gyromultiplication.
$\oplus^{\text{LieAI}}, \oplus^{\text{LE}}, \oplus^{\text{LC}}, \oplus^{\theta\text{-AI}}, \oplus^{\theta\text{-LC}}, \ominus^{\text{LieAI}}$	SPD Lie group operations under AIM, LEM, LCM, and their power-deformed AIM and LCM variants, with $\ominus^{\text{LieAI}}$ denoting the inverse under $\oplus^{\text{LieAI}}$ when operation-specific disambiguation is needed.
$\oplus^{\text{LE}}, \odot^{\text{LE}}, \oplus^{\text{LC}}, \odot^{\text{LC}}$	LEM- and LCM-induced SPD gyrovectors operations, which coincide with linear operations in the corresponding global charts.
BW	Bures–Wasserstein geometry on SPD matrices.
$g^{\text{AI}}, g^{\text{LE}}, g^{\text{LC}}, g^{\theta\text{-E}}, g^{\text{BW}}, g^{\text{CRI}}, g^{\theta\text{-CRI}}$	Metric tensors associated with the corresponding SPD geometries.
$\text{GL}(n), \text{SL}(n)$	The general linear group and the special linear group.
$\text{O}(n)$	The orthogonal group.
$\text{St}(p, n)$	The Stiefel manifold represented by orthonormal frames.
$\text{Gr}(p, n), \widetilde{\text{Gr}}(p, n)$	The Grassmannian manifold represented by orthonormal bases and projection matrices.

$I_{p,n}, \tilde{I}_{p,n}$	Identity points of the Grassmannian under the Orthonormal Basis (ONB) and Projector Perspective (PP) representations.
$\oplus^{\text{Gr}}, \odot^{\text{Gr}}, \ominus^{\text{Gr}}$	Grassmannian gyroaddition, scalar multiplication, and inverse under the ONB representation.
$\tilde{\oplus}^{\text{Gr}}, \tilde{\odot}^{\text{Gr}}, \tilde{\ominus}^{\text{Gr}}$	Grassmannian gyroaddition, scalar multiplication, and inverse under the PP representation.
$\pi : \text{Gr}(p, n) \rightarrow \widetilde{\text{Gr}}(p, n)$	The Grassmannian isometry $\pi(U) = UU^\top$ from ONB to PP.
Matrix commutator	The operation $[A, B] = AB - BA$.
$\text{Cor}^+(n)$	The manifold of full-rank correlation matrices.
$\text{Cor}, \mathcal{I}, \text{inv}$	The correlation normalization map, the cor-inversion operator on correlation matrices, and the matrix inversion operator on SPD matrices.
LT^n	The Euclidean space of $n \times n$ lower triangular matrices.
$\text{LT}^0(n)$	The Euclidean space of $n \times n$ strictly lower triangular matrices.
$\text{LT}^1(n)$	The affine space of $n \times n$ lower triangular matrices with unit diagonal.
$\mathcal{L}^n$	The manifold of Cholesky factors of full-rank correlation matrices, consisting of lower triangular matrices with positive diagonal entries and unit row norm.
$\text{Hol}(n), \text{Row}_0(n)$	The spaces of symmetric hollow matrices and symmetric matrices with null row sum, respectively.
$\Theta, \phi^{\text{EC}}, \text{Log}^\circ, \text{Exp}^\circ, \text{Log}^\star, \text{Exp}^\star$	Diffeomorphisms and charts used by ECM, OLM, and LSM on correlation matrices.
$\mathcal{D}(H), \mathcal{D}^\star(C)$	The diagonal correction operators used in OLM and LSM, respectively.
$\text{Diag}(n), \text{Diag}^+(n)$	The vector space of diagonal matrices and the positive diagonal matrix manifold.
$\mathfrak{S}^n, \mathfrak{S}^\pm(n)$	The permutation group and the signed permutation group acting on correlation matrices.
$(X)_{\text{sym}}, (X)_{\text{sym}+}$	The averaged symmetrization $(X + X^\top)/2$ and the unnormalized symmetrization $X + X^\top$, respectively.

$\text{HS}^i, \text{PHS}^{n-1}, \mathbb{PP}^{n-1}$	The open hemisphere, product hemisphere space, and product of unit Poincaré balls used by PHCM and GyroBN on correlation manifolds.
$\Phi : \text{Cor}^+(n) \rightarrow \mathbb{PP}^{n-1}$	The row-wise Cholesky identification from full-rank correlation matrices to the product of unit Poincaré balls.
$\text{SO}(n), \mathfrak{so}(n)$	The special orthogonal group and its Lie algebra.

Constant-Curvature Manifolds

$\mathcal{M}_K^n$	The K -radius model with curvature parameter K .
$\mathfrak{st}_K^n, \mathbb{D}_K^n$	The K -stereographic model and its positive-curvature projected hypersphere case.
$\mathbb{S}_K^n$	The spherical model space with curvature parameter K .
$\mathbb{S}^n, \mathbb{D}^n$	The unit sphere and unit projected hypersphere. Unit-space notation suppresses the fixed curvature subscript.
$\mathbb{P}_K^n$	The Poincaré ball model of hyperbolic space with curvature $K < 0$.
$\mathbb{P}^n, \mathbb{K}^n, \mathbb{H}^n, \mathbb{L}^n$	The unit Poincaré ball, unit Beltrami–Klein ball, unit hyperboloid, and unit Lorentz reference space. Unit-space notation suppresses the fixed curvature subscript.
$\oplus_K, \ominus_K, \odot_K$	Gyroaddition, gyro-inverse, and scalar gyromultiplication on the K -stereographic model.
λ_x^K	The conformal factor of the K -stereographic model at x .
$\oplus_M, \ominus_M, \odot_M$	Möbius gyroaddition, gyro-inverse, and scalar multiplication on the Poincaré ball.
$\oplus_K^M, \ominus_K^M, \odot_K^M$	Gyroaddition, gyro-inverse, and scalar gyromultiplication on the K -radius model.
$\mathbb{L}_K^n, \mathbb{H}_K^n$	Equivalent notation for the Lorentz, or hyperboloid, model of hyperbolic space with curvature $K < 0$.
$\oplus_L, \ominus_L, \odot_L$	Lorentz gyroaddition, gyro-inverse, and scalar gyromultiplication on the Lorentz model.
$\langle x, y \rangle_{\mathcal{L}}, \ x\ _{\mathcal{L}}$	The Lorentzian inner product and its associated quantity.

$\langle x, y \rangle_K, \ x\ _K$	The curvature-dependent ambient bilinear form on the K -radius model and its induced tangent-space norm. On the $K < 0$ branch, the ambient form is Lorentzian and becomes positive definite only after restriction to a tangent space.
$\mathbb{K}_K^n$	The Beltrami–Klein model of hyperbolic space with curvature $K < 0$.
$\oplus_E, \ominus_E, \odot_E$	Einstein gyroaddition, gyro-inverse, and scalar gyromultiplication on the Beltrami–Klein model.
γ_x	The Einstein gamma factor on the Beltrami–Klein model.
$\tan_K, \sin_K, \cos_K$	Curvature-aware trigonometric functions used for constant-curvature models.
$\mathbb{PV}_K^n$	The Proper Velocity model of hyperbolic space with curvature $K < 0$.
$\oplus_U, \ominus_U, \otimes_U$	PV gyroaddition, gyro-inverse, and scalar gyromultiplication.
β_x, γ_y	The relativistic beta factor on PV space and the gamma factor used by the PV–Poincaré isometry.
$\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}, \pi_{\mathbb{P}_K^n \rightarrow \mathbb{PV}_K^n}$	The mutually inverse isometries between the PV model and the Poincaré ball.
$\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{H}_K^n}, \pi_{\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n}$	The mutually inverse isometries between the PV model and the hyperboloid model.

Chapter 1

Introduction

1.1 Riemannian Deep Learning

Over the past decade or so, *Deep Neural Networks* (DNNs) have achieved significant progress in machine learning [102, 127, 97, 202]. Traditionally, DNNs have been developed under the assumption that the latent geometry of the input data is Euclidean. However, many applications involve non-Euclidean structures, such as manifolds [30, 88]. Therefore, deep learning over Riemannian spaces, referred to as *Riemannian deep learning*, has shown great success in diverse applications, such as computer vision [107, 106, 108, 47, 120], natural language processing [76, 181, 141], graph and knowledge-graph learning [163, 38, 42, 136], multimodal learning [67, 166], recommendation systems [223], signal processing [31, 123], human neuroimaging [167, 134, 135, 233], medical imaging [37], astronomy [44], and genome sequence modeling [119]. Commonly encountered manifolds include special orthogonal groups [165], Symmetric Positive Definite (SPD) manifolds [9], Grassmannian manifolds [71, 19], spherical manifolds [165], and hyperbolic manifolds [173, 33]. Many of these manifolds admit computationally tractable Riemannian operators, including geodesics, exponential and logarithmic maps, and parallel transport.

Building on these geometric tools, several fundamental Euclidean neural network components have been generalized to manifold-valued data, including normalization [31, 34, 142, 123], attention [167], residual blocks [201, 118], classification [76, 159, 161], and Fully Connected (FC) and convolutional layers [106, 107, 108, 76, 181, 45, 161]. However, most existing constructions either remain tied to particular geometries or carry out neural computations in intermediate flat spaces that approximate the intrinsic geometry. Existing normalization methods are either confined to particular SPD met-

rics [31, 123], restricted to matrix Lie groups with a specific distance [34, Sec. 3.2], or applicable more generally but lack theoretical guarantees for controlling sample statistics, as in ManifoldNorm [34, Algs. 1–2] and RBN [142, Alg. 2]. Classification layers frequently map manifold-valued features to tangent Euclidean spaces [106, 31], ambient Euclidean spaces [107], or coordinate Euclidean spaces [36], while intrinsic alternatives may require the generalized law of sines [76] or gyrovector structures [159, 161]. FC and convolutional layers exhibit similar limitations. Early constructions are tailored to SPD, rotation, and Grassmannian manifolds [106, 107, 108], while hyperbolic counterparts rely on tangent-space mappings [76, 147], Lorentz spacetime [45], or Poincaré geometry [181]. The weighted-Fréchet-mean convolution [37] applies more broadly, but constrains the output manifold dimension to match the input dimension. These limitations motivate unified principles that formulate a module once at an appropriate geometric level and then instantiate it across manifolds carrying the required common structure.

Beyond individual modules, deep architectures have been developed for matrix-valued manifolds, including SPD manifolds [106, 37], Grassmannian manifolds [108], and rotation manifolds [107], as well as vector-valued manifolds, including spherical manifolds [13, 182] and hyperbolic manifolds [76, 13, 182, 45, 15]. Nevertheless, existing networks remain concentrated on a limited set of geometric representations and construction tools. Hyperbolic networks, for example, predominantly use the Poincaré ball [76, 181] and Lorentz (hyperboloid) models [45, 15], while alternative models remain less explored [33]. Beyond standard Riemannian and gyrovector operators, Busemann functions and horospheres [29, Ch. II.8] have been incorporated into hyperbolic SVMs [73], hyperbolic PCA [39], Sliced-Wasserstein distances [24], and prototype-learning methods [81]. Despite these developments, their use in neural network design remains limited. Likewise, correlation matrices have received far less attention than SPD covariance representations despite being statistically compact alternatives to covariance matrices [7]. Their recently developed tractable Riemannian geometries [195, 191] indicate a broader design space that remains underexplored.

Finally, both the module and network designs described above ultimately depend on the underlying Riemannian geometry. A Riemannian metric is not merely a means of measuring distance. It provides the theoretical foundation and concrete computational primitives of Riemannian learning algorithms. By assigning inner products to tangent spaces, the metric determines geodesic distances, exponential and logarithmic maps, parallel transport, and Fréchet statistics [165]. These operators enter directly into the design of deep-network components. For example, hyperbolic geometry de-

finer Riemannian classifiers [76, 181], weighted Fréchet means support convolutional layers [37] and normalization [31, 34, 123], and exponential and logarithmic maps underpin attention [167], residual blocks [118], and pooling layers [205]. Consequently, changing the Riemannian metric changes not only the geometry, but also the formulas, parameterizations, computational cost, numerical stability, and ultimately the practical behavior of the associated deep networks. This role is especially evident on the SPD manifold, where a broad range of metrics has been developed [171, 9, 70, 137, 22, 93]. However, most existing metric tensors are fixed, which may limit the expressivity of the induced geometry and its ability to adapt to data or the dynamics of deep networks. Moreover, numerical stability is particularly important in deep-network training, where metric-induced operators are repeatedly evaluated. Developing Riemannian metrics that balance flexibility, tractability, efficiency, and stability is therefore a significant problem for Riemannian deep learning.

1.2 Contributions and Outlines

The contributions of this thesis are organized around three connected perspectives: unified Riemannian module design across manifolds, manifold-specific Riemannian network design, and the design of the underlying Riemannian geometries. The first perspective formulates principled network modules which can be applied to different geometries, including normalization and classification layers. The second perspective addresses cases in which a fully general construction is either intractable or unable to exploit useful manifold-specific structure. It therefore uses the additional structures of particular manifolds to develop manifold-specific modules and network architectures. The third perspective moves from network design under prescribed geometries to the design of the geometry itself, developing flexible, efficient, and numerically stable metrics on SPD manifolds. The contributions and organization of the remaining chapters are summarized below.

Chapter 2 establishes the mathematical foundations used throughout the thesis. The chapter is organized into two parts. The first part develops the general theory required by the subsequent methods, including topology, differential and Riemannian geometry, metric geometry, algebraic structures on manifolds, Riemannian optimization, and matrix functions, and relates these constructions to their Euclidean counterparts. The second part turns to the concrete manifolds studied in later chapters: SPD, full-rank correlation, Grassmannian, and constant-curvature manifolds, together with special orthogonal groups.

Chapter 3 develops unified frameworks for Batch Normalization (BN) across structured classes of manifolds, with the goal of controlling both the Riemannian mean and variance beyond a single manifold or metric. Sec. 3.2 first introduces Lie Group Batch Normalization (LieBN) on Lie groups under invariant metrics. LieBN uses group translations for centering and biasing and tangent-space scaling at the identity for variance control, thereby extending Euclidean BN while preserving the manifold structure. Concrete manifestations are developed on SPD, rotation, and full-rank correlation manifolds, together with efficient implementations and experimental validation. Sec. 3.3 first introduces pseudo-reductive gyrogroups, a new algebraic structure that generalizes classical gyrogroups and Lie groups, thereby providing a more general foundation for principled normalization. Building on this structure, it develops Gyrogroup Batch Normalization (GyroBN), which recovers LieBN as a special case and extends the same normalization principle to manifolds that need not possess a Lie group structure. Finally, GyroBN is instantiated on the Grassmannian, constant-curvature, and full-rank correlation manifolds, and experiments evaluate the resulting layers across matrix-manifold, constant-curvature, and graph learning tasks.

Chapter 4 develops a unified framework for intrinsic classification across Riemannian manifolds. Sec. 4.2 first extends Euclidean Multinomial Logistic Regression (MLR) to SPD manifolds with flat pullback metrics. By formulating classification through the geodesic margin between an SPD input and a decision hyperplane, it derives closed-form classifiers and provides an intrinsic explanation for the widely used LogEig MLR. Sec. 4.3 then replaces the potentially intractable point-to-hyperplane infimum with a Riemannian-trigonometric formulation, extending the classifier beyond flat SPD geometries. The resulting Riemannian Multinomial Logistic Regression (RMLR) requires only an explicit Riemannian logarithmic map, incorporates several existing manifold classifiers as special cases, and yields new instantiations on SPD manifolds under multiple metric families and on the special orthogonal group. Finally, Sec. 4.4 evaluates these classifiers within feedforward, residual, graph, and Lie group networks, as well as in direct manifold-valued classification.

Chapter 5 turns to manifold-specific Riemannian network design, exploiting the additional structures of particular hyperbolic models and correlation manifolds. Sec. 5.2 first introduces the Proper Velocity (PV) model, an unconstrained model of hyperbolic space, and establishes its Riemannian geometry. Based on this geometry, Proper Velocity Neural Networks (PVNNs) develop MLR, FC, convolutional, activation, and normalization layers for stable hyperbolic deep learning. Sec. 5.3 uses Busemann functions and horospheres to develop intrinsic and efficient Busemann Multinomial Logistic

Regression (BMLR) and Busemann Fully Connected (BFC) layers on both the Poincaré and Lorentz models. BMLR interprets its logits through point-to-horosphere distances, while BFC uses the same Busemann logits to construct feature transformations. Sec. 5.4 then develops Correlation Networks (CorNets) for full-rank correlation matrices. It constructs MLR, FC, and convolutional layers under five correlation geometries and derives accurate Riemannian backpropagation. Experiments across vision, graph, and genome learning tasks validate these manifold-specific designs.

Chapter 6 develops flexible, fast, and numerically stable Riemannian metrics on SPD manifolds. Whereas the preceding chapters design modules and networks under prescribed geometries, this chapter designs the underlying metrics. Sec. 6.2 first develops Adaptive Log-Euclidean Metrics (ALEMs). By parameterizing general matrix logarithms within a pullback framework, ALEMs adapt the geometry to data while retaining closed-form Riemannian operators and a compatible abelian Lie group structure; the learned metrics are instantiated in SPD networks and other Riemannian building blocks. Sec. 6.3 then develops product Cholesky geometries, including the Power-Cholesky Metric (PCM) and Bures–Wasserstein–Cholesky Metric (BWCM). These metrics avoid the scalar logarithms and exponentials used by the Log-Cholesky Metric (LCM) and admit fast and stable closed-form Riemannian and algebraic operators. Their effectiveness is evaluated in SPD classification, residual learning, and tensor interpolation, while separate experiments assess computational efficiency, scalability, and numerical stability.

Chapter 7 summarizes the thesis and discusses future directions.

1.3 Summary of Papers Excluded from the Thesis

The technical chapters focus on the core theoretical and methodological contributions. Fourteen additional publications on Riemannian deep learning belong to the same research theme and are summarized below.

- **Riemannian attention.** We first developed self-attention on three specific manifolds: the Grassmannian [210], the full-rank correlation manifold [103], and the SPD manifold under the Bures–Wasserstein geometry [113]. Then, we generalized geometry-specific constructions into a unified attention framework for general matrix manifolds [212].
- **Geometry-aware electroencephalography (EEG) representation learning.** In Li et al. [135], we used hyperbolic embeddings to represent the hierarchical structure of EEG signals and improve cross-domain generalization. In Hu et al.

[104], we introduced Riemannian high-order pooling on SPD manifolds to capture second-order correlations in EEG foundation models.

- **Geometry-specific Riemannian batch normalization.** We developed SPD BN methods under specific Riemannian metrics [214, 215] for stable training.
- **Skeleton-based action recognition.** We developed two SPD-based approaches to skeleton action recognition: a graph convolutional network that uses Gaussian embeddings of high-order skeletal statistics to model inter-subject interactions and global correlations for two-person interaction recognition [216], and an approach that partitions the skeleton into semantic body regions and models their long-range dependencies with SPD representations [213].
- **Hyperbolic neural networks.** In Shi et al. [179], we constructed a fully intrinsic hyperbolic Lorentz neural network whose FC, BN, concatenation, activation, and dropout modules operate within the Lorentz geometry.
- **Hyperbolic multi-view clustering.** In Wang et al. [217], we developed hyperbolic multi-view clustering that aligns view-specific distributions through a hyperbolic sliced-Wasserstein distance while preserving hierarchical semantics in the Lorentz manifold.
- **Riemannian interpretation of global covariance pooling.** In Chen et al. [53], we provided a unified Riemannian interpretation of matrix functions in global covariance pooling, showing that their effectiveness is explained by the Riemannian classifiers they implicitly respect.
- **SPD deep metric learning.** In Wang et al. [211], we combined an SPD network encoder with a Riemannian decoder, local covariance regularization, and deep metric learning to improve image-set classification.

These papers are excluded from this thesis because they extend or apply the core works developed in this thesis. For example, CorAtt [103] applies OLM/LSM correlation geometry to EEG attention, while Sec. 5.4 systematizes how to build neural networks and differentiation over the correlation matrices. GyroAtt [212] adopts the gyro paradigm, following Sec. 3.3. HEEGNet [135] applies the Lorentz GyroBN developed in Sec. 3.3 to domain-specific normalization. CBN [214] and GBWBN [215] specialize Sec. 3.2’s template to specific metrics. ILNN [179] accelerates the Lorentz

GyroBN developed in Sec. 3.3 and follows Sec. 5.2’s point-to-hyperplane design. Finally, RiemGCP [53] applies the RMLR framework developed in Sec. 4.3 to interpret matrix-function normalization as implicit SPD Riemannian classifiers.

1.3. *Summary of Papers Excluded from the Thesis*

Chapter 2

Mathematical Background

2.1 Introduction

This chapter presents the mathematical foundations of the thesis in two parts. The first part develops the general theory and computational tools used throughout the subsequent chapters, beginning with topology and differential geometry [197], proceeding to Riemannian geometry [165], metric geometry [29], and algebraic structures on manifolds [128, 200], and then reviewing Riemannian optimization [1, 27], trivialization [132], and matrix functions and their differentials [21]. Although these concepts can be abstract, many generalize familiar Euclidean constructions, so Euclidean prototypes are provided whenever appropriate. The second part reviews the concrete spaces used later: Symmetric Positive Definite (SPD) and full-rank correlation manifolds, Grassmannian manifolds, special orthogonal groups, and constant-curvature manifolds. Their Riemannian and algebraic operators expose both the common primitives that support unified module design across manifolds and the additional structures later exploited by manifold-specific methods.

2.2 Topology

Topology provides the language for continuity and locality before coordinates are introduced [197, App. A]. This is essential because a manifold is usually not defined by a single global coordinate system as in Euclidean space, but by a family of local coordinate systems.

Recall classical analysis on $\mathbb{R}^n$, where many basic notions are formulated in terms of open sets. In the standard Euclidean setting $\mathbb{R}^n$, openness is defined through open

2.2. Topology

balls: a subset $U \subset \mathbb{R}^n$ is open if, for every $x \in U$, there exists $r > 0$ such that the open ball $B_r(x) = \{y \in \mathbb{R}^n \mid \|y - x\| < r\}$ is contained in U . However, a general abstract space need not carry a norm or a distance, so it may not have a prior notion of open balls. Topology abstracts the essential closure properties of Euclidean open sets, leading to the following definition.

Definition 1 (Topological space [197, Def. A.1]). A *topology* on a set X is a collection $\mathcal{T}$ of subsets of X such that

- (1) $\emptyset \in \mathcal{T}$ and $X \in \mathcal{T}$,
- (2) every union of elements of $\mathcal{T}$ belongs to $\mathcal{T}$, i.e., for any $\{U_\alpha\}_{\alpha \in A} \subset \mathcal{T}$,

$$\bigcup_{\alpha \in A} U_\alpha \in \mathcal{T}. \quad (2.1)$$

- (3) every finite intersection of elements of $\mathcal{T}$ belongs to $\mathcal{T}$, i.e., for any integer $m \geq 1$ and any open sets $U_1, \dots, U_m \in \mathcal{T}$,

$$\bigcap_{i=1}^m U_i \in \mathcal{T}. \quad (2.2)$$

The pair $(X, \mathcal{T})$ is called a *topological space*. Elements of $\mathcal{T}$ are called *open sets*. A subset $F \subset X$ is called *closed* if $X \setminus F$ is open, where $X \setminus F = \{x \in X \mid x \notin F\}$.

A basis records enough open sets to reconstruct the full topology, and second countability controls the size of this local description.

Definition 2 (Basis and second countability [197, Defs. A.6 and A.12]). Let $(X, \mathcal{T})$ be a topological space. A collection $\mathcal{B} \subset \mathcal{T}$ is a *basis* for $\mathcal{T}$ if every open set $U \in \mathcal{T}$ is a union of elements of $\mathcal{B}$. Equivalently, for every $x \in U$ with $U \in \mathcal{T}$, there exists $B \in \mathcal{B}$ such that $x \in B \subset U$. A topological space is *second countable* if its topology has a countable basis.

The next proposition gives the corresponding construction criterion when one starts from a candidate family of subsets rather than from an already specified topology.

Proposition 3 (Criterion [197, Prop. A.8]). *Let X be a set and let $\mathcal{B}$ be a collection of subsets of X . Then $\mathcal{B}$ is a basis for some topology $\mathcal{T}$ on X if and only if*

- (1) X is the union of all sets in $\mathcal{B}$,

(2) whenever $x \in B_1 \cap B_2$ with $B_1, B_2 \in \mathcal{B}$, there exists $B_3 \in \mathcal{B}$ such that $x \in B_3 \subset B_1 \cap B_2$.

In Euclidean space $\mathbb{R}^n$, the family of open balls generates the standard topology. Moreover, $\mathbb{R}^n$ is second countable because the open balls with centers in $\mathbb{Q}^n$ and positive rational radii form a countable basis.

Subspace topology is used whenever a geometric object is described as a subset of an ambient space.

Definition 4 (Subspace topology [197, Sec. A.2]). Let $(X, \mathcal{T})$ be a topological space and let $Y \subset X$. The *subspace topology* on Y is

$$\mathcal{T}_Y = \{Y \cap U \mid U \in \mathcal{T}\}. \quad (2.3)$$

With this topology, Y is called a *subspace* of X .

For Euclidean space, the unit sphere $S^{n-1} = \{x \in \mathbb{R}^n \mid \|x\| = 1\}$ carries the subspace topology inherited from $\mathbb{R}^n$. Its open sets are exactly the sets $S^{n-1} \cap U$, where U is open in $\mathbb{R}^n$.

Definition 5 (Continuous map and homeomorphism [197, Sec. A.7]). Let $(X, \mathcal{T}_X)$ and $(Y, \mathcal{T}_Y)$ be topological spaces. A map $f : X \rightarrow Y$ is *continuous* if $f^{-1}(V) \in \mathcal{T}_X$ for every $V \in \mathcal{T}_Y$. A map $f : X \rightarrow Y$ is a *homeomorphism* if it satisfies:

- (1) f is bijective,
- (2) f and f^{-1} are continuous.

In the Euclidean special case, this definition recovers the familiar notion of continuity from calculus. For a map $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with the standard topologies, f is continuous in the sense of Thm. 5 if and only if, for every $x \in \mathbb{R}^n$ and every $\varepsilon > 0$, there exists $\delta > 0$ such that $\|f(y) - f(x)\| < \varepsilon$ whenever $\|y - x\| < \delta$. Homeomorphic spaces are topologically identical. A simple homeomorphism is the translation $x \mapsto x + a$ on $\mathbb{R}^n$, whose inverse is $x \mapsto x - a$.

For manifolds, one also needs a separation condition that prevents distinct points from being topologically indistinguishable.

Definition 6 (Hausdorff space [197, Def. A.16]). A topological space X is *Hausdorff* if, for any two distinct points $x, y \in X$, there exist disjoint open sets $U, V \subset X$ such that $x \in U$ and $y \in V$.

Topological concept	Euclidean special case
Open and closed sets	$B_r(x) = \{y \in \mathbb{R}^n \mid \ y - x\ < r\},$ $\overline{B}_r(x) = \{y \in \mathbb{R}^n \mid \ y - x\ \leq r\}.$
Basis and second countability	$\mathcal{B}_{\mathbb{Q}} = \{B_r(q) \mid q \in \mathbb{Q}^n, r \in \mathbb{Q}_{>0}\}.$
Subspace topology	$\mathcal{T}_{S^{n-1}} = \{S^{n-1} \cap U \mid U \subset \mathbb{R}^n \text{ open}\}.$
Continuous map	Continuous map in calculus.
Hausdorff space	$x \neq y \Rightarrow B_r(x) \cap B_r(y) = \emptyset$ for $0 < r < \frac{1}{2} \ x - y\ .$

Table 2.1: Euclidean prototypes for topological concepts.

The Hausdorff condition ensures that points can be separated by neighborhoods. For Euclidean space, if $x \neq y$, then the open balls $B_r(x)$ and $B_r(y)$ are disjoint whenever $0 < r < \frac{1}{2} \|x - y\|$. Thus Euclidean space is Hausdorff.

Tab. 2.1 summarizes the Euclidean special cases of the above topological concepts.

2.3 Differential Geometry

In $\mathbb{R}^n$, a basis provides a global coordinate system: every point is represented by a unique coordinate tuple. By contrast, a manifold usually does not admit a single coordinate map covering the entire space. Instead, each point has a neighborhood homeomorphic to an open subset of $\mathbb{R}^n$, and this homeomorphism provides Euclidean coordinates on that neighborhood. This local Euclidean structure permits derivatives, tangent vectors, and smooth maps to be defined intrinsically.

Definition 7 (Topological manifold [197, Sec. 5]). An n -dimensional *topological manifold* is a topological space $\mathcal{M}$ such that

- (1) $\mathcal{M}$ is Hausdorff,
- (2) $\mathcal{M}$ is second countable,
- (3) every point $p \in \mathcal{M}$ has a neighborhood U homeomorphic to an open subset of $\mathbb{R}^n$.

The integer n is called the *dimension* of $\mathcal{M}$.

The Hausdorff and second-countability assumptions exclude pathological spaces and ensure that local coordinates behave like ordinary Euclidean neighborhoods.

Charts are local coordinate systems, and an atlas is a collection of such coordinate systems covering the whole manifold.

Definition 8 (Chart and atlas [197, Sec. 5]). Let $\mathcal{M}$ be an n -dimensional topological manifold. A *chart* on $\mathcal{M}$ is a pair (U, φ) satisfying:

- (1) $U \subset \mathcal{M}$ is open,
- (2) $\varphi : U \rightarrow \varphi(U) \subset \mathbb{R}^n$ is a homeomorphism onto an open subset of $\mathbb{R}^n$.

Writing $\varphi = (x^1, \dots, x^n)$, the functions $x^i : U \rightarrow \mathbb{R}$ are called the *coordinate functions* of the chart. Equivalently, if $r^i : \mathbb{R}^n \rightarrow \mathbb{R}$ denotes the i -th standard coordinate projection, then $x^i = r^i \circ \varphi$. An *atlas* is a collection of charts $\mathcal{A} = \{(U_\alpha, \varphi_\alpha)\}_{\alpha \in A}$ satisfying

$$\mathcal{M} = \bigcup_{\alpha \in A} U_\alpha. \quad (2.4)$$

To do calculus consistently across charts, changes of coordinates must preserve smoothness.

Definition 9 (Smooth atlas and smooth manifold [197, Sec. 5]). Two charts (U, φ) and (V, ψ) on $\mathcal{M}$ are *smoothly compatible* if the following transition maps are smooth maps between open subsets of Euclidean spaces:

- (1) $\psi \circ \varphi^{-1} : \varphi(U \cap V) \rightarrow \psi(U \cap V)$,
- (2) $\varphi \circ \psi^{-1} : \psi(U \cap V) \rightarrow \varphi(U \cap V)$.

A *smooth atlas* is an atlas whose charts are pairwise smoothly compatible. A *smooth manifold* is a topological manifold equipped with a maximal smooth atlas.

Smooth compatibility means that changing coordinates does not destroy differentiability. This is the formal mechanism that allows a derivative computed in one coordinate chart to represent an intrinsic geometric object.

Smooth maps between manifolds are defined by checking their coordinate representations.

Definition 10 (Smooth map and diffeomorphism [197, Sec. 6]). Let $\mathcal{M}$ and $\mathcal{N}$ be smooth manifolds. A map $f : \mathcal{M} \rightarrow \mathcal{N}$ is *smooth* if, for every $p \in \mathcal{M}$, every chart (U, φ) around p , and every chart (V, ψ) around $f(p)$ with $f(U) \subset V$, the coordinate representation

$$\psi \circ f \circ \varphi^{-1} : \varphi(U) \rightarrow \psi(V) \quad (2.5)$$

is smooth. A smooth map $f : \mathcal{M} \rightarrow \mathcal{N}$ is a *diffeomorphism* if it satisfies:

- (1) f is bijective,

(2) $f^{-1} : \mathcal{N} \rightarrow \mathcal{M}$ is smooth.

Diffeomorphisms refine homeomorphisms by preserving smooth structure, so diffeomorphic manifolds are identical from the viewpoint of smooth geometry.

Before defining tangent spaces on manifolds, it is useful to recall what a tangent vector does in Euclidean calculus. Fix $x \in \mathbb{R}^n$ and a direction $v \in \mathbb{R}^n$. The directional derivative at x in the direction v can be viewed as an operator on smooth functions,

$$D_{x,v} : C^\infty(\mathbb{R}^n) \rightarrow \mathbb{R}, \quad D_{x,v}(f) = \left. \frac{d}{dt} \right|_{t=0} f(x + tv). \quad (2.6)$$

This operator is linear in f and, by the ordinary product rule, satisfies

$$D_{x,v}(fh) = f(x)D_{x,v}(h) + h(x)D_{x,v}(f), \quad f, h \in C^\infty(\mathbb{R}^n). \quad (2.7)$$

Thus a Euclidean tangent vector can be recognized not only as an arrow v , but also as a first-order operator that differentiates smooth functions at x . The derivation viewpoint keeps precisely this algebraic behavior and extends it to manifolds, where no global vector structure is available.

Definition 11 (Tangent space [197, Sec. 8]). Let $\mathcal{M}$ be a smooth manifold and let $p \in \mathcal{M}$. A *derivation at p* is a map $v : C^\infty(\mathcal{M}) \rightarrow \mathbb{R}$ satisfying:^a

(1) linearity,

$$v(af + bh) = av(f) + bv(h), \quad a, b \in \mathbb{R}, \quad f, h \in C^\infty(\mathcal{M}), \quad (2.8)$$

(2) the Leibniz rule,

$$v(fh) = f(p)v(h) + h(p)v(f), \quad f, h \in C^\infty(\mathcal{M}). \quad (2.9)$$

The set of all derivations at p is a vector space, called the *tangent space* of $\mathcal{M}$ at p and denoted by $T_p\mathcal{M}$.

^aStrictly speaking, the domain should be the algebra of *germs* of smooth functions at p , namely equivalence classes of smooth functions that agree on some neighborhood of p . For simplicity, we write $C^\infty(\mathcal{M})$.

The tangent space $T_p\mathcal{M}$ is a vector space whose operations are pointwise addition and scalar multiplication of derivations. Let $(U, \varphi) = (U, x^1, \dots, x^n)$ be a chart containing p . The chart induces the *coordinate tangent vector* $\left. \frac{\partial}{\partial x^i} \right|_p$, which is the derivation at

p defined by

$$\left. \frac{\partial}{\partial x^i} \right|_p (f) = \frac{\partial (f \circ \varphi^{-1})}{\partial x^i} (\varphi(p)), \quad f \in C^\infty(\mathcal{M}), \quad i = 1, \dots, n. \quad (2.10)$$

Thus $\left. \frac{\partial}{\partial x^i} \right|_p$ differentiates f in the i -th Euclidean coordinate direction after f is expressed in the chart. The coordinate tangent vectors $\left. \frac{\partial}{\partial x^1} \right|_p, \dots, \left. \frac{\partial}{\partial x^n} \right|_p$ form a basis of $T_p\mathcal{M}$ [197, Prop. 8.9]. Consequently, every $v \in T_p\mathcal{M}$ has a unique coordinate representation

$$v = \sum_{i=1}^n v^i \left. \frac{\partial}{\partial x^i} \right|_p, \quad v^i \in \mathbb{R}, \quad (2.11)$$

so this coordinate basis identifies $T_p\mathcal{M}$ with $\mathbb{R}^n$.

The differential of a smooth map generalizes the Jacobian matrix.

Definition 12 (Differential [197, Sec. 8]). Let $f : \mathcal{M} \rightarrow \mathcal{N}$ be a smooth map and let $p \in \mathcal{M}$. The *differential* of f at p is the linear map $f_{*,p} : T_p\mathcal{M} \rightarrow T_{f(p)}\mathcal{N}$ defined by

$$(f_{*,p}(v))(h) = v(h \circ f), \quad v \in T_p\mathcal{M}, \quad h \in C^\infty(\mathcal{N}). \quad (2.12)$$

Throughout this chapter, we write the differential as $f_{*,p}$ and its action on v as $f_{*,p}(v)$. Later computations also use the notation $d_p f(v)$.

The intrinsic differential recovers the ordinary Jacobian in local coordinates. Let $(U, \varphi) = (U, x^1, \dots, x^n)$ be a chart around $p \in \mathcal{M}$ and let $(V, \psi) = (V, y^1, \dots, y^m)$ be a chart around $f(p) \in \mathcal{N}$. The differential $f_{*,p}$ is

$$\begin{aligned} & f_{*,p} \left(\left[\left. \frac{\partial}{\partial x^1} \right|_p \quad \cdots \quad \left. \frac{\partial}{\partial x^n} \right|_p \right] \right) \\ &= \left[\left. \frac{\partial}{\partial y^1} \right|_{f(p)} \quad \cdots \quad \left. \frac{\partial}{\partial y^m} \right|_{f(p)} \right] \begin{bmatrix} \frac{\partial (y^1 \circ f)}{\partial x^1}(p) & \cdots & \frac{\partial (y^1 \circ f)}{\partial x^n}(p) \\ \vdots & \ddots & \vdots \\ \frac{\partial (y^m \circ f)}{\partial x^1}(p) & \cdots & \frac{\partial (y^m \circ f)}{\partial x^n}(p) \end{bmatrix}. \end{aligned} \quad (2.13)$$

The (i, j) -th entry of this matrix is $\partial (y^i \circ f) / \partial x^j(p)$ for $i = 1, \dots, m$ and $j = 1, \dots, n$. Hence, $f_{*,p}$ is represented by the Jacobian matrix of the local coordinate representation $\psi \circ f \circ \varphi^{-1}$ evaluated at $\varphi(p)$ [197, Prop. 8.11].

The differential gives an intrinsic definition of the velocity of a curve.

Definition 13 (Smooth curve and velocity [197, Sec. 8.6]). Let $I \subset \mathbb{R}$ be an open interval and let $c : I \rightarrow \mathcal{M}$ be a smooth curve. The *velocity vector* of c at $t_0 \in I$ is

$$c'(t_0) := c_{*,t_0} \left(\frac{d}{dt} \Big|_{t=t_0} \right) \in T_{c(t_0)}\mathcal{M}. \quad (2.14)$$

Equivalently, for every $h \in C^\infty(\mathcal{M})$,

$$c'(t_0)(h) = \frac{d}{dt} \Big|_{t=t_0} h(c(t)). \quad (2.15)$$

Proposition 14 (Velocity in local coordinates [197, Prop. 8.15]). Let $c : I \rightarrow \mathcal{M}$ be a smooth curve and let $(U, \varphi) = (U, x^1, \dots, x^n)$ be a chart containing $c(t)$. Then the velocity is obtained by differentiating the coordinate functions:

$$c'(t) = \sum_{i=1}^n \frac{d(x^i \circ c)}{dt}(t) \frac{\partial}{\partial x^i} \Big|_{c(t)}. \quad (2.16)$$

Thus the coefficients of $c'(t)$ in the chart-induced basis are precisely the ordinary derivatives of the coordinate representation $\varphi \circ c$.

Every smooth curve through p yields a tangent vector in $T_p\mathcal{M}$, and conversely every tangent vector is the velocity of some smooth curve through p [197, Prop. 8.16]. Therefore,

$$T_p\mathcal{M} = \{c'(0) \mid \varepsilon > 0, \quad c : (-\varepsilon, \varepsilon) \rightarrow \mathcal{M} \text{ is smooth}, \quad c(0) = p\}. \quad (2.17)$$

For any curve c representing $v \in T_p\mathcal{M}$ in this way, the derivation v acts as the directional derivative $v(h) = \frac{d}{dt} \Big|_{t=0} h(c(t))$ [197, Prop. 8.17].

Curves also provide a classical method for computing differentials.

Proposition 15 (Differential via curves [197, Prop. 8.18]). Let $F : \mathcal{M} \rightarrow \mathcal{N}$ be a smooth map, let $p \in \mathcal{M}$, and let $v \in T_p\mathcal{M}$. If $c : (-\varepsilon, \varepsilon) \rightarrow \mathcal{M}$ is any smooth curve satisfying $c(0) = p$ and $c'(0) = v$, then the differential maps the velocity of c to the velocity of its image curve:

$$F_{*,p}(v) = (F \circ c)'(0) = \frac{d}{dt} \Big|_{t=0} F(c(t)). \quad (2.18)$$

In particular, this expression is independent of the chosen representative curve.

The following examples illustrate the above proposition in the vector and matrix cases.

Example 16 (Sphere). Let $\mathbb{S}^n = \{y \in \mathbb{R}^{n+1} \mid \|y\| = 1\}$ and consider the normalization map $\nu : \mathbb{R}^{n+1} \setminus \{0\} \rightarrow \mathbb{S}^n$ defined by $\nu(x) = x/\|x\|$. For $x \neq 0$ and $v \in \mathbb{R}^{n+1}$, the curve $c(t) = x + tv$ has initial velocity v . Applying Eq. (2.18) gives

$$\nu_{*,x}(v) = \frac{d}{dt} \Big|_{t=0} \frac{x + tv}{\|x + tv\|} = \frac{1}{\|x\|} \left(v - \frac{\langle x, v \rangle}{\|x\|^2} x \right) \in T_{\nu(x)} \mathbb{S}^n. \quad (2.19)$$

In particular, when $x \in \mathbb{S}^n$, the differential is $\nu_{*,x}(v) = v - \langle x, v \rangle x$, the orthogonal projection of v onto $T_x \mathbb{S}^n$.

Example 17 (General linear group). Let $\mathrm{GL}(n) = \{A \in \mathbb{R}^{n \times n} \mid \det(A) \neq 0\}$ denote the manifold of invertible real $n \times n$ matrices. For $G \in \mathrm{GL}(n)$, define $L_G : \mathrm{GL}(n) \rightarrow \mathrm{GL}(n)$ by $L_G(B) = GB$. Since $\mathrm{GL}(n)$ is an open submanifold of $\mathbb{R}^{n \times n}$, its tangent spaces are identified with $\mathbb{R}^{n \times n}$. For $X \in T_{I_n} \mathrm{GL}(n) \cong \mathbb{R}^{n \times n}$, the curve $c(t) = I_n + tX$ remains in $\mathrm{GL}(n)$ for sufficiently small t and satisfies $c'(0) = X$. Hence,

$$(L_G)_{*,I_n}(X) = \frac{d}{dt} \Big|_{t=0} G(I_n + tX) = GX, \quad (2.20)$$

so the differential of left multiplication is again left multiplication [197, Ex. 8.19].

A vector field assigns a tangent vector to each point in a smooth way.

Definition 18 (Vector field [197, Def. 12.7]). A *vector field* on a smooth manifold $\mathcal{M}$ is a map $X : \mathcal{M} \rightarrow T\mathcal{M}$ such that $X(p) \in T_p \mathcal{M}$ for every $p \in \mathcal{M}$. It is *smooth* if, for every smooth function $f \in C^\infty(\mathcal{M})$, the function $p \mapsto X(p)f$ is smooth.

Tab. 2.2 summarizes the Euclidean special cases of the above differential-geometric concepts.

2.4 Riemannian Geometry

Riemannian geometry equips each tangent space with an inner product. This turns local tangent vectors into measurable directions and induces global geometric objects

Differential-geometric concept	Euclidean special case
Smooth manifolds	$\mathbb{R}^n$.
Charts	$\varphi = \text{id}_{\mathbb{R}^n}$.
Smooth map	Smooth map in calculus.
Tangent space	$T_x \mathbb{R}^n \cong \mathbb{R}^n$.

Table 2.2: Euclidean prototypes for differential-geometric concepts.

such as lengths and distances.

Definition 19 (Riemannian manifold [165, Defs. 3.1 and 3.2]). Let $\mathcal{M}$ be a smooth manifold. A *Riemannian metric* on $\mathcal{M}$ is a smooth assignment

$$p \mapsto g_p : T_p \mathcal{M} \times T_p \mathcal{M} \rightarrow \mathbb{R} \quad (2.21)$$

such that, for every $p \in \mathcal{M}$, g_p satisfies:

- (1) g_p is bilinear,
- (2) $g_p(v, w) = g_p(w, v)$ for all $v, w \in T_p \mathcal{M}$,
- (3) $g_p(v, v) > 0$ for all nonzero $v \in T_p \mathcal{M}$.

The pair $(\mathcal{M}, g)$ is called a *Riemannian manifold*. We write the value of the metric as $\langle v, w \rangle_p = g_p(v, w)$ for $v, w \in T_p \mathcal{M}$. The induced norm is

$$\|v\|_p = \sqrt{\langle v, v \rangle_p}, \quad v \in T_p \mathcal{M}. \quad (2.22)$$

The Riemannian metric extends the Euclidean inner product to curved spaces. Unless explicitly stated otherwise, a *manifold* means a Riemannian manifold, and the pair $(\mathcal{M}, g)$ is abbreviated as $\mathcal{M}$. We also use $\langle v, w \rangle_p$ for the Riemannian metric.

Once each tangent vector has a norm, one can measure the length of curves and define the induced shortest-path distance.

Definition 20 (Length and geodesic distance [165, Defs. 5.11 and 5.15]). Let $\mathcal{M}$ be a manifold and let $\alpha : [a, b] \rightarrow \mathcal{M}$ be a piecewise smooth curve. The *length* of α is

$$L(\alpha) = \int_a^b \|\dot{\alpha}(t)\|_{\alpha(t)} dt. \quad (2.23)$$

The *geodesic distance* between $x, y \in \mathcal{M}$ is

$$d_{\mathcal{M}}(x, y) = \inf_{\alpha} L(\alpha), \quad (2.24)$$

where the infimum is taken over all piecewise smooth curves $\alpha : [a, b] \rightarrow \mathcal{M}$ satisfying $\alpha(a) = x$ and $\alpha(b) = y$.

The geodesic distance in $\mathbb{R}^n$ is exactly the straight-line distance. On a connected manifold $\mathcal{M}$, the distance function $d_{\mathcal{M}} : \mathcal{M} \times \mathcal{M} \rightarrow [0, \infty)$ is a metric, which makes $(\mathcal{M}, d_{\mathcal{M}})$ a metric space [165, Prop. 5.18].

The notion of a connection generalizes the directional derivative of tangent vectors to curved spaces. It is used to define geodesics, parallel transport, and curvature.

Definition 21 (Connection and covariant derivative [165, Def. 3.9]). Let $\mathfrak{X}(\mathcal{M})$ denote the space of smooth vector fields on $\mathcal{M}$. A *connection* on $\mathcal{M}$ is a map

$$D : \mathfrak{X}(\mathcal{M}) \times \mathfrak{X}(\mathcal{M}) \rightarrow \mathfrak{X}(\mathcal{M}), \quad (X, Y) \mapsto D_X Y, \quad (2.25)$$

that satisfies, for all $X, Y, Z \in \mathfrak{X}(\mathcal{M})$, $f_1, f_2, f \in C^\infty(\mathcal{M})$, and $a, b \in \mathbb{R}$:

(1) $C^\infty(\mathcal{M})$ -linearity in the first argument,

$$D_{f_1 X + f_2 Y} Z = f_1 D_X Z + f_2 D_Y Z, \quad (2.26)$$

(2) $\mathbb{R}$ -linearity in the second argument,

$$D_X(aY + bZ) = aD_X Y + bD_X Z, \quad (2.27)$$

(3) the Leibniz rule,

$$D_X(fY) = X(f)Y + fD_X Y. \quad (2.28)$$

The vector field $D_X Y$ is called the *covariant derivative* of Y in the direction X .

A Riemannian metric determines a canonical connection by requiring zero torsion and compatibility with the metric.

Theorem 22 (Levi-Civita connection [165, Thm. 3.11]). *Every Riemannian manifold $(\mathcal{M}, g)$ admits a unique connection D satisfying, for all smooth vector fields $X, Y, Z \in \mathfrak{X}(\mathcal{M})$:*

(1) *torsion-freeness*,

$$D_X Y - D_Y X = [X, Y], \quad (2.29)$$

(2) *compatibility with the metric*,

$$X(\langle Y, Z \rangle) = \langle D_X Y, Z \rangle + \langle Y, D_X Z \rangle. \quad (2.30)$$

This connection is called the Levi-Civita connection.

Example 23 (Euclidean Levi-Civita connection [165, Def. 3.8 and Lem. 3.14]).

Let $x^1, \dots, x^n$ be the standard coordinates on $\mathbb{R}^n$, and let

$$X = \sum_{i=1}^n X^i \frac{\partial}{\partial x^i}, \quad Y = \sum_{j=1}^n Y^j \frac{\partial}{\partial x^j} \quad (2.31)$$

be smooth vector fields. The Levi-Civita connection of the Euclidean metric is

$$D_X Y = \sum_{j=1}^n X(Y^j) \frac{\partial}{\partial x^j} = \sum_{i,j=1}^n X^i \frac{\partial Y^j}{\partial x^i} \frac{\partial}{\partial x^j}. \quad (2.32)$$

Thus, $D_X Y$ is the ordinary directional derivative of the component functions of Y along X . In particular, the standard coordinate vector fields are parallel:

$$D_{\frac{\partial}{\partial x^i}} \frac{\partial}{\partial x^j} = 0, \quad (2.33)$$

for all i, j .

The Levi-Civita connection is the canonical connection of Riemannian geometry. It determines geodesics, parallel transport, and curvature. It also induces a covariant derivative for vector fields along a curve.

Proposition 24 (Induced covariant derivative [165, Prop. 3.18]). *Let $\mathcal{M}$ be a manifold with Levi-Civita connection D , let $\alpha : I \rightarrow \mathcal{M}$ be a smooth curve, and let $\mathfrak{X}(\alpha)$ denote the space of smooth vector fields along α . There is a unique map*

$$\mathfrak{X}(\alpha) \rightarrow \mathfrak{X}(\alpha), \quad V \mapsto V' = \frac{D}{dt}(V), \quad (2.34)$$

called the induced covariant derivative along α , satisfying:

- (1) $(aV + bW)' = aV' + bW'$ for $V, W \in \mathfrak{X}(\alpha)$ and $a, b \in \mathbb{R}$.
- (2) $(fV)' = \frac{df}{dt}V + fV'$ for $V \in \mathfrak{X}(\alpha)$ and $f \in C^\infty(I)$.
- (3) Let $U \subseteq \mathcal{M}$ be an open neighborhood of $\alpha(I)$, and let $Y \in \mathfrak{X}(U)$ be a smooth vector field on U . If $V \in \mathfrak{X}(\alpha)$ is the vector field along α obtained by restricting Y to the curve, namely $V(t) = Y_{\alpha(t)} \in T_{\alpha(t)}\mathcal{M}$, then

$$V'(t) = D_{\dot{\alpha}(t)}Y, \quad (2.35)$$

where the right-hand side denotes the covariant derivative of the field Y .

- (4) For $V, W \in \mathfrak{X}(\alpha)$,

$$\frac{d}{dt} \langle V, W \rangle_{\alpha(t)} = \langle V', W \rangle_{\alpha(t)} + \langle V, W' \rangle_{\alpha(t)}. \quad (2.36)$$

Geodesics are the Riemannian counterparts of straight lines.

Definition 25 (Geodesic [165, Ch. 3]). Let $\mathcal{M}$ be a manifold with Levi-Civita connection D . A smooth curve $\gamma : I \rightarrow \mathcal{M}$ is a *geodesic* if

$$\frac{D\dot{\gamma}}{dt} = D_{\dot{\gamma}}\dot{\gamma} = 0. \quad (2.37)$$

Consequently, every geodesic has *constant speed*: $\|\dot{\gamma}(t)\|_{\gamma(t)}$ is constant on I .

Unlike straight lines in Euclidean space, geodesics on a general manifold need not be globally length-minimizing. They are locally length-minimizing curves [165, Lem. 5.14 and Prop. 5.16].

Geodesics also turn tangent vectors into manifold points through the exponential map, and locally turn nearby manifold points back into tangent vectors through the logarithmic map.

Definition 26 (Exponential and logarithmic maps [165, Def. 3.29 and Prop. 3.30]). Let $\mathcal{M}$ be a manifold. For $x \in \mathcal{M}$ and $v \in T_x\mathcal{M}$, let $\gamma_{x,v}$ be the geodesic satisfying $\gamma_{x,v}(0) = x$ and $\dot{\gamma}_{x,v}(0) = v$. The *Riemannian exponential map* at x is

$$\text{Exp}_x(v) = \gamma_{x,v}(1), \quad (2.38)$$

where it is defined. As Exp_x is locally invertible around $0 \in T_x\mathcal{M}$, its local inverse

is called the *Riemannian logarithmic map* at x and is denoted by Log_x .

This pair is used repeatedly in Riemannian neural networks to move between non-linear manifold-valued data and linear tangent-space computations.

To compare or aggregate tangent vectors based at different manifold points, tangent vectors must be transported along curves.

Definition 27 (Parallel transport [165, Prop. 3.19]). Let $\alpha : [a, b] \rightarrow \mathcal{M}$ be a smooth curve and let $V(t)$ be a vector field along α . The field V is *parallel along* α if

$$\frac{DV}{dt} = 0 \quad (2.39)$$

for all $t \in [a, b]$. Given $v \in T_{\alpha(a)}\mathcal{M}$, the endpoint $V(b) \in T_{\alpha(b)}\mathcal{M}$ of the unique parallel vector field satisfying $V(a) = v$ is called the *parallel transport* of v along α .

When the connecting curve is the relevant geodesic from x to y , we denote parallel transport by $\text{PT}_{x \rightarrow y} : T_x\mathcal{M} \rightarrow T_y\mathcal{M}$.

Proposition 28 (Parallel transport is an isometry [165, Lem. 3.20]). Let $V(t)$ and $W(t)$ be parallel vector fields along a smooth curve $\alpha : [a, b] \rightarrow \mathcal{M}$. Then

$$\langle V(t), W(t) \rangle_{\alpha(t)} \quad (2.40)$$

is constant in t . Consequently, parallel transport along α defines a linear isometry between tangent spaces.

The connection also measures the failure of second covariant derivatives to commute, which is encoded by the curvature tensor.

Definition 29 (Curvature [165, Lem. 3.35]). Let $\mathcal{M}$ be a manifold with Levi-Civita connection D . The *Riemannian curvature tensor* is the map

$$R : \mathfrak{X}(\mathcal{M}) \times \mathfrak{X}(\mathcal{M}) \times \mathfrak{X}(\mathcal{M}) \rightarrow \mathfrak{X}(\mathcal{M}) \quad (2.41)$$

defined by

$$R(X, Y)Z = D_X D_Y Z - D_Y D_X Z - D_{[X, Y]}Z, \quad (2.42)$$

where $[X, Y]$ denotes the Lie bracket of vector fields:

$$[X, Y](f) = X(Y(f)) - Y(X(f)), \quad f \in C^\infty(\mathcal{M}). \quad (2.43)$$

Sectional curvature extracts from the curvature tensor the curvature of each two-dimensional tangent plane.

Definition 30 (Sectional curvature [165, Lem. 3.39]). Let $\mathcal{M}$ be a manifold with curvature tensor R . For a two-dimensional subspace $\sigma \subset T_x\mathcal{M}$ spanned by linearly independent vectors $u, v \in T_x\mathcal{M}$, the *sectional curvature* of σ is

$$K_x(\sigma) = \frac{\langle R(u, v)v, u \rangle_x}{\langle u, u \rangle_x \langle v, v \rangle_x - \langle u, v \rangle_x^2}. \quad (2.44)$$

It is independent of the choice of basis $\{u, v\}$ for σ .

Positive, zero, and negative curvature correspond to spherical, Euclidean, and hyperbolic behavior, which will be discussed later.

A central consequence of non-positive curvature is that the exponential map can become globally invertible under suitable assumptions.

Theorem 31 (Cartan–Hadamard theorem [165, Thm. 10.22]). *Let $\mathcal{M}$ be a complete, connected, and simply connected manifold whose sectional curvature is everywhere non-positive. Then, for every $x \in \mathcal{M}$, the exponential map $\text{Exp}_x : T_x\mathcal{M} \rightarrow \mathcal{M}$ is a global diffeomorphism.*

This theorem explains why several geometries in machine learning are algorithmically convenient, as their exponential maps allow tangent-space computations to be performed globally on the manifold.

The distance also supports averaging manifold-valued data through Fréchet means.

Definition 32 (Fréchet mean and variance [171, Sec. 2]). Let $(\mathcal{X}, d)$ be a metric space and let $x_1, \dots, x_N \in \mathcal{X}$ with weights $w_i > 0$ satisfying $\sum_{i=1}^N w_i = 1$. A *weighted Fréchet mean* is any minimizer

$$\text{WFM}(\{w_i\}, \{x_i\}) \in \arg\min_{z \in \mathcal{X}} \sum_{i=1}^N w_i d(x_i, z)^2. \quad (2.45)$$

When $w_i = \frac{1}{N}$ for all i , it is called the *Fréchet mean*, and we write $\text{FM}(\{x_i\})$ for

Geometric concept	Euclidean special case
Riemannian metric	$\langle u, v \rangle_x = \langle u, v \rangle.$
Geodesic distance	$d(x, y) = \ x - y\ .$
Geodesic	$\gamma(t) = (1 - t)x + ty.$
Exponential map	$\text{Exp}_x(v) = x + v.$
Logarithmic map	$\text{Log}_x(y) = y - x.$
Parallel transport	$\text{PT}_{x \rightarrow y} = \text{id}_{\mathbb{R}^n}.$
Sectional curvature	$K \equiv 0$ on $\mathbb{R}^n.$
Weighted Fréchet mean	$\text{WFM}(\{w_i\}, \{x_i\}) = \sum_{i=1}^N w_i x_i.$
Fréchet variance	$\text{Var} = \min_z \sum_{i=1}^N w_i \ x_i - z\ ^2.$

Table 2.3: Euclidean prototypes for Riemannian-geometric concepts.

the corresponding minimizer. The infimum of the objective is called the *Fréchet variance*.

The objective above is defined on any metric space. On a manifold, the distance is usually the geodesic distance. If the data lie in a sufficiently small geodesic ball, the weighted Fréchet mean exists and is unique [2, Thm. 2.1].

Pullback metrics allow a complex manifold to inherit a computationally convenient geometry from a simpler prototype space.

Definition 33 (Pullback and Riemannian isometry [165, Defs. 2.8 and 3.6]). Let $\mathcal{M}$ and $\mathcal{N}$ be smooth manifolds, let g be a Riemannian metric on $\mathcal{N}$, and let $f : \mathcal{M} \rightarrow \mathcal{N}$ be smooth. The *pullback* of g by f is the symmetric tensor field f^*g on $\mathcal{M}$ defined by

$$(f^*g)_p(v, w) = g_{f(p)}(f_{*,p}(v), f_{*,p}(w)), \quad p \in \mathcal{M}, \quad v, w \in T_p\mathcal{M}. \quad (2.46)$$

If f^*g is positive definite at every point, it is a Riemannian metric on $\mathcal{M}$. In particular, when f is a diffeomorphism and $\mathcal{M}$ is equipped with the pullback metric f^*g , the map $f : (\mathcal{M}, f^*g) \rightarrow (\mathcal{N}, g)$ is a *Riemannian isometry*.

A Riemannian isometry is the Riemannian version of a diffeomorphism: it preserves the metric, lengths, geodesic distances, geodesics, exponential maps, logarithmic maps, parallel transport, curvature, and distance-based quantities such as Fréchet means and variances [165, Ch. 3].

Tab. 2.3 summarizes the Euclidean special cases of the above geometric concepts.

2.5 Metric Geometry

Metric geometry extends Riemannian geometry to the more general setting of metric spaces, where no differentiable structure is assumed.¹ The theory develops geodesics and curvature without smooth structure, providing the metric tools used later for hyperbolic neural layers. We begin by recalling basic notions in metric spaces.

Definition 34 (Metric space [29, Def. I.1.1]). A *metric space* is a pair $(\mathcal{X}, d)$ where $\mathcal{X}$ is a nonempty set and $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ satisfies, for all $x, y, z \in \mathcal{X}$:

- (1) $d(x, y) \geq 0$ and $d(x, y) = 0$ if and only if $x = y$,
- (2) $d(x, y) = d(y, x)$,
- (3) $d(x, z) \leq d(x, y) + d(y, z)$.

A connected Riemannian manifold becomes a metric space when equipped with its geodesic distance [165, Prop. 5.18]. The metric-space perspective abstracts away coordinates and focuses on distances, geodesics, and comparison geometry.

Geodesics, rays, and lines generalize unit-speed minimizing geodesics to metric spaces.

Definition 35 (Geodesic, geodesic ray, and geodesic line [29, Ch. I.1]). Let $(\mathcal{X}, d)$ be a metric space. A *geodesic* joining x to y is a continuous map $\gamma : [0, l] \rightarrow \mathcal{X}$ with $\gamma(0) = x$ and $\gamma(l) = y$ such that

$$d(\gamma(t), \gamma(t')) = |t - t'|, \quad t, t' \in [0, l]. \quad (2.47)$$

A *geodesic ray* is a continuous map $\gamma : [0, \infty) \rightarrow \mathcal{X}$ such that $d(\gamma(t), \gamma(t')) = |t - t'|$ for all $t, t' \geq 0$. A *geodesic line* is a continuous map $\gamma : \mathbb{R} \rightarrow \mathcal{X}$ such that $d(\gamma(t), \gamma(t')) = |t - t'|$ for all $t, t' \in \mathbb{R}$.

Geodesic metric spaces abstract the requirement that every pair of points be joined by a distance-realizing geodesic.

Definition 36 (Geodesic metric space [29, Ch. I.1]). The metric space $(\mathcal{X}, d)$ is a *geodesic metric space*, or more briefly a *geodesic space*, if every pair of points in $\mathcal{X}$ is joined by a geodesic. It is *uniquely geodesic* if there is exactly one geodesic

¹Here, the missing differentiable structure refers specifically to a smooth manifold structure. Metric analogues of geodesics and curvature can still be defined without it.

joining x to y for all $x, y \in \mathcal{X}$.

Convexity is defined through geodesics, generalizing linear convexity in Euclidean space and geodesic convexity on manifolds.

Definition 37 (Convex subset [29, Ch. I.1]). Let $(\mathcal{X}, d)$ be a metric space. A subset $C \subseteq \mathcal{X}$ is *convex* if every pair $x, y \in C$ can be joined by a geodesic in $\mathcal{X}$ and the image of every such geodesic is contained in C .

We next review concepts that extend curvature from manifolds to metric spaces. The reference spaces are the model spaces of constant curvature.

Definition 38 (Model space [29, Ch. I.2]). For $K \in \mathbb{R}$, the *model space* (M_K^n, d_K) is given by

$$(M_K^n, d_K) = \begin{cases} (\mathbb{S}^n, \frac{1}{\sqrt{K}}d), & K > 0, \\ (\mathbb{R}^n, d), & K = 0, \\ (\mathbb{L}^n, \frac{1}{\sqrt{-K}}d), & K < 0, \end{cases} \quad (2.48)$$

where d is the geodesic distance in the corresponding manifold. The unit sphere $\mathbb{S}^n$ and unit Lorentz manifold $\mathbb{L}^n$ are reviewed in Sec. 2.9.5. The diameter of M_K^2 is denoted by

$$D_K = \begin{cases} \pi/\sqrt{K}, & K > 0, \\ \infty, & K \leq 0. \end{cases} \quad (2.49)$$

Definition 39 (Comparison triangle [29, Lem. I.2.14 and Sec. II.1]). Let $(\mathcal{X}, d)$ be a geodesic metric space and let $\triangle(x, y, z)$ be a geodesic triangle in $\mathcal{X}$ with side lengths $a = d(y, z)$, $b = d(x, z)$, and $c = d(x, y)$. A *comparison triangle* for $\triangle(x, y, z)$ in M_K^2 is a triangle $\triangle(\bar{x}, \bar{y}, \bar{z})$ such that $d_K(\bar{y}, \bar{z}) = a$, $d_K(\bar{x}, \bar{z}) = b$, and $d_K(\bar{x}, \bar{y}) = c$. When $a + b + c < 2D_K$, the comparison triangle exists. If p lies on the side from x to y , then a *comparison point* for p is the point $\bar{p}$ on the side from $\bar{x}$ to $\bar{y}$ such that $d(x, p) = d_K(\bar{x}, \bar{p})$ and $d(y, p) = d_K(\bar{y}, \bar{p})$. Comparison points on the other two sides are defined analogously.

CAT(K) spaces encode curvature through triangle comparison with the model plane M_K^2 . Intuitively, a CAT(K) space is a metric space whose triangles are thinner than the corresponding comparison triangles in M_K^2 .

Definition 40 (CAT(K) space [29, Def. II.1.1]). Let $(\mathcal{X}, d)$ be a metric space and let $K \in \mathbb{R}$. Let Δ be a geodesic triangle in $\mathcal{X}$ with perimeter less than $2D_K$, and let $\bar{\Delta} \subset M_K^2$ be a comparison triangle for Δ . The triangle Δ satisfies the CAT(K) inequality if, for all $p, q \in \Delta$ and all corresponding comparison points $\bar{p}, \bar{q} \in \bar{\Delta}$,

$$d(p, q) \leq d_K(\bar{p}, \bar{q}). \quad (2.50)$$

Then $\mathcal{X}$ is called a CAT(K) space as follows:

- (1) If $K \leq 0$, then $\mathcal{X}$ is a CAT(K) space if $\mathcal{X}$ is a geodesic space all of whose geodesic triangles satisfy the CAT(K) inequality.
- (2) If $K > 0$, then $\mathcal{X}$ is a CAT(K) space if $\mathcal{X}$ is D_K -geodesic and all geodesic triangles in $\mathcal{X}$ of perimeter less than $2D_K$ satisfy the CAT(K) inequality.

Here, D_K -geodesic means that for every pair of points $x, y \in \mathcal{X}$ with $d(x, y) < D_K$, there is a geodesic joining x to y .

Definition 41 (Hadamard space [29, p. 159]). A *Hadamard space* is a complete CAT(0) space.

In particular, a Hadamard manifold, a complete, simply connected Riemannian manifold with non-positive sectional curvature, is a Hadamard space [29, Thm. II.4.1].

Proposition 42 (Orthogonal projection [29, Prop. II.2.4]). Let $(\mathcal{X}, d)$ be a CAT(0) space and let $C \subseteq \mathcal{X}$ be a convex subset that is complete in the induced metric. For every $x \in \mathcal{X}$, there exists a unique point $\pi_C(x) \in C$ such that

$$d(x, \pi_C(x)) = \inf_{y \in C} d(x, y) = d(x, C). \quad (2.51)$$

If x' belongs to the geodesic segment $[x, \pi_C(x)]$, then

$$\pi_C(x') = \pi_C(x). \quad (2.52)$$

The map $\pi_C : \mathcal{X} \rightarrow C$ is called the orthogonal projection, or simply the projection.

With geodesics, one can define asymptotic rays, boundary points, Busemann functions, horoballs, and horospheres in metric spaces.

Definition 43 (Asymptotic rays and boundary points [29, Def. II.8.1]). Let $(\mathcal{X}, d)$ be a metric space. Two geodesic rays $\gamma, \eta : [0, \infty) \rightarrow \mathcal{X}$ are *asymptotic* if

$$\text{there exists } C \geq 0 \text{ such that } d(\gamma(t), \eta(t)) \leq C \text{ for all } t \geq 0. \quad (2.53)$$

The set $\partial\mathcal{X}$ of *boundary points*, also called *points at infinity* or *ideal points*, is the set of equivalence classes of geodesic rays, where two geodesic rays are equivalent if and only if they are asymptotic.

Asymptotic rays generalize parallel lines. In hyperbolic geometry, they point toward the same ideal boundary point, which becomes a direction for defining Busemann functions.

Definition 44 (Busemann function, horoball, and horosphere [29, Def. II.8.17]). Let $(\mathcal{X}, d)$ be a metric space and let $\gamma : [0, \infty) \rightarrow \mathcal{X}$ be a geodesic ray. If the limit exists, the *Busemann function* associated with γ is

$$B^\gamma(x) = \lim_{t \rightarrow \infty} (d(x, \gamma(t)) - t), \quad x \in \mathcal{X}. \quad (2.54)$$

For $\tau \in \mathbb{R}$, the sublevel set

$$HB_\tau^\gamma = \{x \in \mathcal{X} \mid B^\gamma(x) \leq \tau\} \quad (2.55)$$

is a *horoball*, and the level set

$$H_\tau^\gamma = \{x \in \mathcal{X} \mid B^\gamma(x) = \tau\} \quad (2.56)$$

is a *horosphere*.

In Hadamard spaces, the Busemann limit exists [29, Lem. II.8.18]. Moreover, Busemann functions associated with asymptotic rays agree up to an additive constant.

Corollary 45 (Busemann functions of asymptotic rays [29, Cor. II.8.20]). *If $\mathcal{X}$ is a Hadamard space, then the Busemann functions associated with asymptotic rays in $\mathcal{X}$ are equal up to addition of a constant.*

In Euclidean space, if $\gamma(t) = tv$ with $\|v\| = 1$, then $B^\gamma(x) = -\langle x, v \rangle$. Thus the Busemann function provides an intrinsic generalization of the Euclidean inner product, up to sign. Therefore, horospheres generalize Euclidean hyperplanes. Tab. 2.4

Metric-geometric concept	Euclidean special case
Geodesic, geodesic ray, and geodesic line	Straight segment, ray, and line.
Asymptotic geodesic rays	Parallel rays.
Busemann function $-B^\gamma(x)$	Inner product $\langle x, v \rangle$.
Horosphere H_τ^γ	Hyperplane.
Horospheres associated with asymptotic rays	Parallel hyperplanes.

Table 2.4: Euclidean prototypes for metric-geometric concepts.

summarizes the Euclidean special cases of the metric-geometric concepts.

2.6 Algebraic Structures on Manifolds

Euclidean neural networks rely on vector addition and scalar multiplication to combine, translate, and rescale features. Manifold-valued representations generally lack a global linear structure, which motivates algebraic structures on manifolds that play analogous roles. This section reviews groups, Lie groups, gyrogroups and gyrovector spaces, which generalize addition, subtraction, and scalar multiplication to curved spaces.

Definition 46 (Group [128, Ch. I.2]). A *group* is a nonempty set G equipped with a binary operation^a $\oplus : G \times G \rightarrow G$ satisfying, for all $x, y, z \in G$:

- (1) associativity, $x \oplus (y \oplus z) = (x \oplus y) \oplus z$,
- (2) identity, there exists $e \in G$, called the *identity element* or *neutral element*, such that $e \oplus x = x \oplus e = x$,
- (3) inverse, for every $x \in G$, there exists $\ominus x \in G$ such that $(\ominus x) \oplus x = x \oplus (\ominus x) = e$.

^aThe group operation is usually written multiplicatively and called multiplication, as it is usually noncommutative. For a commutative group, it is often written additively and called addition. In this thesis, we use $\oplus$ for simplicity.

Definition 47 (Abelian group [128, Ch. I.2]). A group $(G, \oplus)$ is an *abelian group*, or *commutative group*, if $x \oplus y = y \oplus x$ for all $x, y \in G$.

Lie groups are simultaneously algebraic and geometric objects. They add smooth structure to groups and require the group operations to be smooth.

Definition 48 (Lie group [197, Def. 6.20]). A *Lie group* is a smooth manifold G equipped with a binary operation $\oplus : G \times G \rightarrow G$ satisfying:

- (1) $(G, \oplus)$ is a group,
- (2) the multiplication map $(x, y) \mapsto x \oplus y$ is smooth,
- (3) the inversion map $x \mapsto \ominus x$ is smooth.

Definition 49 (Abelian Lie group). A Lie group $(G, \oplus)$ is an *abelian Lie group*, or *commutative Lie group*, if its underlying group is abelian, namely $x \oplus y = y \oplus x$ for all $x, y \in G$.

Example 50 (Matrix Lie groups). The general linear group $\text{GL}(n)$ introduced in Thm. 17 is a Lie group under matrix multiplication. Standard matrix Lie subgroups of $\text{GL}(n)$ include the *special linear group*, the *orthogonal group*, and the *special orthogonal group*:

$$\begin{aligned} \text{SL}(n) &= \{A \in \text{GL}(n) \mid \det(A) = 1\}, \\ \text{O}(n) &= \{A \in \text{GL}(n) \mid A^\top A = I_n\}, \\ \text{SO}(n) &= \text{O}(n) \cap \text{SL}(n). \end{aligned} \tag{2.57}$$

Gyrogroups relax the associativity axiom of groups. The deviation from associativity is controlled by gyrations through the gyroassociative law.

Definition 51 (Gyrogroup [200, Def. 2.7]). Given a nonempty set G with a binary operation $\oplus : G \times G \rightarrow G$, $(G, \oplus)$ forms a gyrogroup if its binary operation satisfies the following axioms for any $x, y, z \in G$:

- (1) Left identity: there exists at least one element $e \in G$, called a left identity or neutral element, such that $e \oplus x = x$.
- (2) Left inverse: there exists an element $\ominus x \in G$, called a left inverse of x , such that $\ominus x \oplus x = e$.
- (3) Left gyroassociative law: there exists an automorphism $\text{gyr}[x, y] : G \rightarrow G$ for each $x, y \in G$ such that

$$x \oplus (y \oplus z) = (x \oplus y) \oplus \text{gyr}[x, y]z. \tag{2.58}$$

The automorphism $\text{gyr}[x, y]$ is called the gyroautomorphism, or the gyration

of G generated by x, y .

(4) Left reduction law:

$$\text{gyr}[x, y] = \text{gyr}[x \oplus y, y]. \quad (2.59)$$

If every gyration is the identity map, the gyroassociative law reduces to ordinary associativity. Hence, gyrogroups naturally generalize groups.

Definition 52 (Gyrocommutative gyrogroup [200, Def. 2.8]). A gyrogroup $(G, \oplus)$ is gyrocommutative if it satisfies

$$x \oplus y = \text{gyr}[x, y](y \oplus x) \quad (\text{gyrocommutative law}). \quad (2.60)$$

Gyrocommutativity replaces commutativity. It says that exchanging two operands is possible after applying the appropriate gyration. Similarly, a gyrovector space generalizes a vector space.

Definition 53 (Gyrovector space [54]). A gyrocommutative gyrogroup $(G, \oplus)$ equipped with a scalar gyromultiplication $\odot : \mathbb{R} \times G \rightarrow G$ is called a *gyrovector space* if it satisfies the following axioms for $s, t \in \mathbb{R}$ and $x, y, z \in G$:

(1) Identity scalar multiplication:

$$1 \odot x = x. \quad (2.61)$$

(2) Scalar distributive law:

$$(s + t) \odot x = s \odot x \oplus t \odot x. \quad (2.62)$$

(3) Scalar associative law:

$$(st) \odot x = s \odot (t \odot x). \quad (2.63)$$

(4) Gyroautomorphism:

$$\text{gyr}[x, y](t \odot z) = t \odot \text{gyr}[x, y]z. \quad (2.64)$$

(5) Identity gyroautomorphism:

$$\text{gyr}[s \odot x, t \odot x] = \text{id}, \quad (2.65)$$

where id is the identity map.

Remark 54. Nguyen [157, Def. 2.3] presented a similar definition, except that the identity scalar multiplication axiom also includes $0 \odot x = t \odot e = e$ and $(-1) \odot x = \ominus x$. As implied by Ungar [200, Thm. 6.4], these conditions are redundant.

Just as an inner product space augments a vector space with a compatible inner product, a real inner product gyrovector space augments a gyrovector space with an ambient inner product and corresponding compatibility axioms.

Definition 55 (Real inner product gyrovector space [200, Def. 6.2]). Let $(G, \oplus, \odot)$ be a gyrovector space and let $\langle \cdot, \cdot \rangle$ denote the Euclidean inner product on $\mathbb{R}^n$ with associated norm $\|\cdot\|$. We call $(G, \oplus, \odot, \langle \cdot, \cdot \rangle)$ a *real inner product gyrovector space* if the following conditions hold.

- (1) $G \subseteq \mathbb{R}^n$ and inherits the inner product $\langle \cdot, \cdot \rangle$ and norm $\|\cdot\|$.
- (2) Inner product gyroinvariance:

$$\langle \text{gyr}[x, y]u, \text{gyr}[x, y]v \rangle = \langle u, v \rangle, \quad \forall x, y, u, v \in G. \quad (2.66)$$

- (3) Scaling property:

$$\frac{|s| \odot x}{\|s \odot x\|} = \frac{x}{\|x\|}, \quad \forall x \in G \setminus \{0\}, \quad \forall s \in \mathbb{R} \setminus \{0\}. \quad (2.67)$$

- (4) Let $\|G\| = \{\pm \|x\| \mid x \in G\} \subset \mathbb{R}$. The set $\|G\|$ forms a one-dimensional real vector space with respect to the vector addition and scalar multiplication induced by $\oplus$ and $\odot$ on G .

- (5) Homogeneity property:

$$\|s \odot x\| = |s| \odot \|x\|, \quad \forall x \in G, \quad \forall s \in \mathbb{R}. \quad (2.68)$$

- (6) Gyrotriangle inequality:

$$\|x \oplus y\| \leq \|x\| \oplus \|y\|, \quad \forall x, y \in G. \quad (2.69)$$

Definition 56 (Gyrovector space isomorphisms [200, Def. 6.89]). Let

$$(G_1, \oplus_1, \odot_1) \quad \text{and} \quad (G_2, \oplus_2, \odot_2) \quad (2.70)$$

be real inner product gyrovector spaces. A map $\phi : G_1 \rightarrow G_2$ is a *gyrovector space isomorphism* if it is bijective and satisfies

$$\phi(x \oplus_1 y) = \phi(x) \oplus_2 \phi(y), \quad \forall x, y \in G_1, \quad (2.71)$$

$$\phi(t \odot_1 x) = t \odot_2 \phi(x), \quad \forall x \in G_1, \forall t \in \mathbb{R}, \quad (2.72)$$

and preserves the inner product of unit gyrovectors,

$$\frac{\langle \phi(x), \phi(y) \rangle}{\|\phi(x)\| \|\phi(y)\|} = \frac{\langle x, y \rangle}{\|x\| \|y\|}, \quad \forall x, y \in G_1 \text{ with } x \neq \mathbf{0}, y \neq \mathbf{0}. \quad (2.73)$$

A useful property is that gyrovector space isomorphisms preserve the gyration, inverse, and identity.

Proposition 57. [↓] Let $(G_1, \oplus_1, \odot_1)$ and $(G_2, \oplus_2, \odot_2)$ be real inner product gyrovector spaces with gyrations gyr_1 and gyr_2 , respectively. If $\phi : G_1 \rightarrow G_2$ is a gyrovector space isomorphism, then for all $x, y, z \in G_1$,

$$\phi(\text{gyr}_1[x, y]z) = \text{gyr}_2[\phi(x), \phi(y)]\phi(z), \quad (2.74)$$

$$\phi(e_1) = e_2, \quad (2.75)$$

$$\phi(\ominus_1 x) = \ominus_2 \phi(x), \quad (2.76)$$

where e_1 and e_2 are the gyro identities in G_1 and G_2 , respectively.

Intuitively, a gyrovector space generalizes a vector space to curved spaces. The gyro-operations associated with a manifold $\mathcal{M}$ can be defined as follows. Given a predefined origin $e \in \mathcal{M}$, we assume that the relevant exponential maps, logarithmic maps, and parallel transports are well-defined. Following Nguyen and Yang [159, Eqs. (1)–(3)], for $x, y, z \in \mathcal{M}$ and $t \in \mathbb{R}$, these operations are defined as

$$x \oplus y = \text{Exp}_x(\text{PT}_{e \rightarrow x}(\text{Log}_e(y))), \quad (2.77)$$

$$t \odot x = \text{Exp}_e(t \text{Log}_e(x)), \quad (2.78)$$

$$\ominus x = \text{Exp}_e(-\text{Log}_e(x)), \quad (2.79)$$

$$\text{gyr}[x, y]z = (\ominus(x \oplus y)) \oplus (x \oplus (y \oplus z)). \quad (2.80)$$

The corresponding *gyro inner product*, *gyronorm*, and *gyrodistance* are

$$\langle x, y \rangle_{\text{gyr}} = \langle \text{Log}_e(x), \text{Log}_e(y) \rangle_e, \quad (2.81)$$

$$\|x\|_{\text{gyr}} = \sqrt{\langle x, x \rangle_{\text{gyr}}}, \quad (2.82)$$

$$d_{\text{gyr}}(x, y) = \|\ominus x \oplus y\|_{\text{gyr}}. \quad (2.83)$$

If the above operations satisfy the axioms in Thm. 53, then $(\mathcal{M}, \oplus, \ominus)$ is a gyrovector space. In Euclidean space, the above definitions reduce to the ordinary vector-space structure.

2.7 Riemannian Optimization

This section briefly reviews Riemannian optimization, which addresses the following manifold-constrained optimization problems:

$$\min_{w \in \mathcal{M}} f(w). \quad (2.84)$$

Here, $\mathcal{M}$ is a Riemannian manifold and $f : \mathcal{M} \rightarrow \mathbb{R}$ is a smooth objective function.

Riemannian gradient [165, Def. 3.47]. The *Riemannian gradient* of f at $w \in \mathcal{M}$, denoted by $\text{grad}_w f \in T_w \mathcal{M}$, is the unique tangent vector satisfying

$$\langle \text{grad}_w f, v \rangle_w = f_{*,w}(v), \quad \forall v \in T_w \mathcal{M}. \quad (2.85)$$

Eq. (2.85) characterizes the Riemannian gradient through the directional derivative $f_{*,w}(v)$. This is the same characterization as in Euclidean space, where the Euclidean gradient satisfies $f_{*,w}(v) = \langle \nabla_w f, v \rangle$ for $w, v \in \mathbb{R}^n$. As in $\mathbb{R}^n$, the Cauchy–Schwarz inequality shows that, among all unit tangent directions, the directional derivative is minimized by $-\text{grad}_w f / \|\text{grad}_w f\|_w$ whenever $\text{grad}_w f \neq 0$. Therefore, $-\text{grad}_w f$ is the direction of steepest descent.

Riemannian optimizers. To solve the objective in Eq. (2.84), we take *Stochastic Gradient Descent (SGD)* [175] as an example to show how to generalize a Euclidean optimizer to the Riemannian setting [1]. At iteration t , let f_t denote a stochastic or mini-batch objective associated with f . When $\mathcal{M} = \mathbb{R}^m$, Euclidean SGD updates an

iterate $w^{(t)} \in \mathbb{R}^m$ as

$$w^{(t+1)} = w^{(t)} - \alpha_t \nabla_{w^{(t)}} f_t, \quad (2.86)$$

where $\alpha_t > 0$ is the learning rate and $\nabla_{w^{(t)}} f_t$ is the Euclidean gradient of f_t . On a general manifold, the descent direction must belong to the tangent space at the current iterate, and ordinary vector addition cannot return that direction to the manifold. The corresponding *Riemannian Stochastic Gradient Descent (RSGD)* [25, Sec. 2.3] update is

$$w^{(t+1)} = \text{Exp}_{w^{(t)}}(-\alpha_t \text{grad}_{w^{(t)}} f_t), \quad (2.87)$$

whenever the exponential map is defined for this step. In practical algorithms, a retraction can replace the exact exponential map by a first-order approximation [27, Def. 3.47 and Sec. 4.3], but a systematic treatment of retractions is beyond the scope of this chapter. A widely used package is the **Geoopt** [125], which integrates manifold-valued parameters and Riemannian optimizers into PyTorch, supporting RSGD and adaptive optimizers such as Riemannian Adam [18]. In-depth discussions of Riemannian optimization can be found in Absil et al. [1], Boumal [27].

Trivialization. Apart from solving Eq. (2.84) directly on the manifold, another approach is to parameterize manifold-valued variables with Euclidean parameters and thereby use Euclidean optimization, which is called *trivialization* [132]. Let $\phi : \mathbb{R}^m \rightarrow \mathcal{M}$ be a smooth surjective map, let $z \in \mathbb{R}^m$ be a trainable Euclidean parameter, and set $w = \phi(z) \in \mathcal{M}$. The constrained objective can then be written as

$$\min_{z \in \mathbb{R}^m} f(\phi(z)). \quad (2.88)$$

In practice, the map ϕ could be the exponential map or retraction.

2.8 Matrix Functions

This section reviews some matrix functions widely used on matrix manifolds.

2.8.1 Matrix Functions and Differentials

We denote the Euclidean space of $n \times n$ real symmetric matrices by $\mathcal{S}^n$ and the SPD manifold of $n \times n$ SPD matrices by $\mathcal{S}_{++}^n$. Let $\mathring{I}$ be an open interval of $\mathbb{R}$ and let $f : \mathring{I} \rightarrow \mathbb{R}$ be a smooth function. For any symmetric matrix S whose eigenvalues lie in

$\mathring{I}$, the associated symmetric matrix function is defined by

$$f : S \mapsto Uf(\Sigma)U^\top \in \mathcal{S}^n, \text{ with } S = U\Sigma U^\top \text{ as the eigendecomposition.} \quad (2.89)$$

Its differential is known as the Daleckii–Krein formula:

$$f_{*,S}(V) = U (L_f \otimes (U^\top V U)) U^\top, \quad \forall V \in \mathcal{S}^n, \quad (2.90)$$

$$[L_f]_{i,j} = \begin{cases} \frac{f(\sigma_i) - f(\sigma_j)}{\sigma_i - \sigma_j}, & \text{if } \sigma_i \neq \sigma_j \\ f'(\sigma_i), & \text{otherwise} \end{cases} \quad (2.91)$$

where L_f is called the Loewner matrix, its (i, j) -th entry is defined in Eq. (2.91), and $\otimes$ denotes the Hadamard product. Three special cases are the matrix logarithm $\log : \mathcal{S}_{++}^n \rightarrow \mathcal{S}^n$, the matrix exponential $\exp : \mathcal{S}^n \rightarrow \mathcal{S}_{++}^n$, which is its inverse, and the matrix power map $P_\theta : \mathcal{S}_{++}^n \rightarrow \mathcal{S}_{++}^n$ defined by $P_\theta(P) = P^\theta$. See Bhatia [20, Eqs. (2.38)–(2.40)] or Bhatia [21, Thm. V.3.3] for more details.

Let $\mathcal{L}_{++}^n$ be the Cholesky space of lower triangular matrices with positive diagonal entries. The Cholesky map is $\text{Chol} : \mathcal{S}_{++}^n \rightarrow \mathcal{L}_{++}^n$ with inverse $\text{Chol}^{-1}(L) = LL^\top$. Let $P \in \mathcal{S}_{++}^n$, $L = \text{Chol}(P)$, $V \in T_P \mathcal{S}_{++}^n$, and $X \in T_L \mathcal{L}_{++}^n$. For any square matrix A , set $A_{1/2} = [A] + \frac{1}{2}\mathbb{D}(A)$, where $[A]$ is the strictly lower triangular part and $\mathbb{D}(A)$ is the diagonal matrix formed from the diagonal of A . As shown by Lin [137, Prop. 4], the differentials of Chol and Chol^{-1} are

$$\text{Chol}_{*,P}(V) = L (L^{-1} V L^{-\top})_{1/2}, \quad \text{Chol}_{*,L}^{-1}(X) = XL^\top + LX^\top. \quad (2.92)$$

2.8.2 Backpropagation Through Matrix Functions

Symmetric matrix functions. The above symmetric matrix functions can be backpropagated using the Daleckii–Krein formula. Although PyTorch supports automatic differentiation through eigendecomposition [169], its backward pass requires computing $(\sigma_i - \sigma_j)^{-1}$ [110, Prop. 1], which may trigger numerical instability when two eigenvalues approach each other. Following Brooks et al. [31, Eq. (13)], we instead use the Daleckii–Krein expression in Eqs. (2.90) and (2.91) for the backward pass [21, Thm. V.3.3]. As shown in Eq. (2.91), the divided difference converges to the derivative $f'(\sigma_i)$ when two eigenvalues approach each other, making the expression numerically more stable.

Cholesky decomposition. The backpropagation of the Cholesky decomposition has been studied by Murray [154]. In the experiments, we use `torch.linalg.cholesky`

and its automatic differentiation.

2.9 Example Manifolds

This section collects the concrete model spaces that recur in later chapters. Each manifold is first described by its underlying set and representation, and then summarized through the Riemannian and algebraic operators.

2.9.1 Symmetric Positive Definite Manifolds

The SPD manifold has shown great success in diverse applications [106, 37, 205, 141, 48, 123]. We denote the set of $n \times n$ SPD matrices by $\mathcal{S}_{++}^n$ and the vector space of $n \times n$ real symmetric matrices by $\mathcal{S}^n$. As shown by Arsigny et al. [9], $\mathcal{S}_{++}^n$ forms an open submanifold of the Euclidean space $\mathcal{S}^n$. We review five common Riemannian metrics on $\mathcal{S}_{++}^n$: the *affine-invariant metric (AIM)* [171], the *log-Euclidean metric (LEM)* [9], the *power-Euclidean metric (PEM)* [70], the *log-Cholesky metric (LCM)* [137], and the *Bures–Wasserstein metric (BWM)* [22]. LEM, AIM, and PEM are represented by the parameterized families (α, β) -LEM, (α, β) -AIM, and (θ, α, β) -EM, respectively. Their common (α, β) parameters refer to the following $O(n)$ -invariant inner product on $\mathcal{S}^n$ [196]:

$$\langle A, B \rangle^{(\alpha, \beta)} = \alpha \langle A, B \rangle + \beta \operatorname{tr}(A) \operatorname{tr}(B), \quad (2.93)$$

where $A, B \in \mathcal{S}^n$ and $(\alpha, \beta) \in \mathbf{ST} = \{(\alpha, \beta) \in \mathbb{R}^2 \mid \min(\alpha, \alpha + n\beta) > 0\}$. The standard LEM and AIM are recovered from (α, β) -LEM and (α, β) -AIM, respectively, at $(\alpha, \beta) = (1, 0)$. Likewise, the standard θ -PEM is recovered from (θ, α, β) -EM at $(\alpha, \beta) = (1, 0)$. The (θ, α, β) -EM formulas below assume $\theta \neq 0$; their limit as $\theta \rightarrow 0$ is (α, β) -LEM.

Tabs. 2.5 and 2.6 summarize the associated operators under these five geometries with the following notation.

- (1) **General notation.** Let $P, Q \in \mathcal{S}_{++}^n$ be SPD matrices, let $V, W \in T_P \mathcal{S}_{++}^n$ be tangent vectors, and let $\{P_i\}_{i=1}^N \subset \mathcal{S}_{++}^n$ be a dataset. The norms induced by $\langle \cdot, \cdot \rangle^{(\alpha, \beta)}$ and the standard Frobenius inner product $\langle \cdot, \cdot \rangle$ are denoted by $\|\cdot\|^{(\alpha, \beta)}$ and $\|\cdot\|$, respectively.²

²We use $\|\cdot\|$ for both the Frobenius norm of matrices and the ℓ_2 norm of vectors, since both norms are induced by the standard Euclidean inner product on the ambient Euclidean space.

- (2) **Symmetric matrix functions.** Symmetric matrix functions and their differentials are reviewed in Sec. 2.8.1. In addition to the matrix logarithm and exponential recorded there, the SPD operator tables use the matrix power map $P_\theta(P) = P^\theta$, whose differential is obtained by specializing Eq. (2.90).
- (3) **LCM.** The Cholesky map, its inverse, the half-diagonal operator, and their differentials are reviewed in Sec. 2.8.1. Let $L = \text{Chol}(P)$ and $K = \text{Chol}(Q)$. In the LCM column, the corresponding tangent vectors are $X = \text{Chol}_{*,P}(V)$ and $Y = \text{Chol}_{*,P}(W)$. We define the Log-Cholesky map by $\psi_{\text{LC}}(P) = \lfloor L \rfloor + \text{Dlog}(\mathbb{D}(L))$, where $\text{Dlog}(\cdot)$ is the diagonal element-wise logarithm. The diagonal matrices $\mathbb{K}$, $\mathbb{L}$, $\mathbb{X}$, and $\mathbb{Y}$ contain the diagonal entries of K , L , X , and Y , respectively.
- (4) **Gyrovector.** The three geometries in Tab. 2.5 also admit gyrovector spaces. As shown by Nguyen [157, Sec. 3.1] and Nguyen and Yang [159, Sec. 2.4], AIM, LEM, and LCM each induce a gyro-structure via Eqs. (2.77) to (2.83). For LEM and LCM, the gyrovector spaces reduce to vector spaces:

$$\begin{aligned} P \oplus^{\text{LE}} Q &= \exp(\log(P) + \log(Q)), & t \odot^{\text{LE}} P &= \exp(t \log(P)), \\ P \oplus^{\text{LC}} Q &= \psi_{\text{LC}}^{-1}(\psi_{\text{LC}}(P) + \psi_{\text{LC}}(Q)), & t \odot^{\text{LC}} P &= \psi_{\text{LC}}^{-1}(t \psi_{\text{LC}}(P)). \end{aligned} \quad (2.94)$$

The above two vector additions are the Lie group operations in Tab. 2.5. For AIM, the gyroaddition and scalar gyromultiplication are

$$P \oplus^{\text{AI}} Q = P^{1/2} Q P^{1/2}, \quad t \odot^{\text{AI}} P = P^t. \quad (2.95)$$

In particular, the AIM gyroaddition above is not the AIM Lie group operation; the latter is listed in Tab. 2.5.

- (5) **BWM.** The Lyapunov operator $\mathcal{L}_P[V]$ is defined by

$$\mathcal{L}_P[V]P + P\mathcal{L}_P[V] = V. \quad (2.96)$$

For BWM parallel transport, we only present the case where P and Q are commuting matrices, namely $P = U\Sigma U^\top$ and $Q = U\Delta U^\top$, with $\Sigma = \text{diag}(\sigma_i)$ and $\Delta = \text{diag}(\delta_i)$.

- (6) **Completeness.** BWM and (θ, α, β) -EM with $\theta \neq 0$ are geodesically incomplete, meaning that their exponential maps are not defined globally. For BWM, the exponential map is locally defined only for tangent vectors satisfying $\mathcal{L}_P[V] + I_n \in$

Metric	(α, β) -LEM	(α, β) -AIM	LCM
$Q \oplus P$	$\exp(\log(P) + \log(Q))$	$KPK^\top$	$\text{Chol}^{-1}([L + K] + \mathbb{K}\mathbb{L})$
$g_P(V, W)$	$\langle \log_{*,P}(V), \log_{*,P}(W) \rangle^{(\alpha, \beta)}$	$\langle P^{-1}V, WP^{-1} \rangle^{(\alpha, \beta)}$	$\langle [X], [Y] \rangle + \langle \mathbb{X}\mathbb{L}^{-1}, \mathbb{Y}\mathbb{L}^{-1} \rangle$
$d(P, Q)$	$\ \log(P) - \log(Q)\ ^{(\alpha, \beta)}$	$\ \log(Q^{-1/2}PQ^{-1/2})\ ^{(\alpha, \beta)}$	$\ \psi_{\text{LC}}(P) - \psi_{\text{LC}}(Q)\ $
$\text{FM}(\{P_i\})$	$\exp\left(\frac{1}{N} \sum_{i=1}^N \log(P_i)\right)$	Karcher flow	$\psi_{\text{LC}}^{-1}\left(\frac{1}{N} \sum_{i=1}^N \psi_{\text{LC}}(P_i)\right)$
$\text{Log}_P Q$	$(\log_{*,P})^{-1}[\log(Q) - \log(P)]$	$P^{1/2} \log(P^{-1/2}QP^{-1/2})P^{1/2}$	$\text{Chol}_{*,L}^{-1}([K] - [L] + \mathbb{L} \text{Dlog}(\mathbb{L}^{-1}\mathbb{K}))$
$\text{Exp}_P V$	$\exp(\log(P) + \log_{*,P}(V))$	$P^{1/2} \exp(P^{-1/2}VP^{-1/2})P^{1/2}$	$\text{Chol}^{-1}([L] + [X] + \mathbb{L} \exp(\mathbb{X}\mathbb{L}^{-1}))$
$\gamma(t; P, Q)$	$\exp[\log(P) + t(\log(Q) - \log(P))]$	$P^{1/2}(P^{-1/2}QP^{-1/2})^t P^{1/2}$	$\text{Chol}^{-1}\left\{[L] + t([K] - [L]) + \frac{\mathbb{K}t}{\mathbb{L}^{t-1}}\right\}$
References	[9, 196]; Thm. 147	[171, 196]	[137]; Thm. 147

 Table 2.5: Lie group structures and associated Riemannian operators on $\mathcal{S}_{++}^n$.

Metric	(θ, α, β) -EM	BWM
$g_P(V, W)$	$\frac{1}{\theta^2} \langle (P_\theta)_{*,P}(V), (P_\theta)_{*,P}(W) \rangle^{(\alpha, \beta)}$	$\frac{1}{2} \langle \mathcal{L}_P[V], W \rangle$
$d(P, Q)$	$\frac{1}{ \theta } \ Q^\theta - P^\theta\ ^{(\alpha, \beta)}$	$[\text{tr}(P) + \text{tr}(Q) - 2 \text{tr}((PQ)^{1/2})]^{1/2}$
$\text{Log}_P Q$	$[(P_\theta)_{*,P}]^{-1}(Q^\theta - P^\theta)$	$(PQ)^{1/2} + (QP)^{1/2} - 2P$
$\text{PT}_{P \rightarrow Q}(V)$	$[(P_\theta)_{*,Q}]^{-1} \circ (P_\theta)_{*,P}(V)$	$U \left[\sqrt{\frac{\delta_i + \delta_j}{\sigma_i + \sigma_j}} [U^\top V U]_{ij} \right] U^\top$
$\text{Exp}_P V$	$(P^\theta + (P_\theta)_{*,P}(V))^{1/\theta}$	$P + V + \mathcal{L}_P[V]P\mathcal{L}_P[V]$
$\gamma(t; P, Q)$	$((1-t)P^\theta + tQ^\theta)^{1/\theta}$	$A_t A_t^\top$ $A_t = (1-t)P^{1/2} + tP^{-1/2}(P^{1/2}QP^{1/2})^{1/2}$
References	[70, 196]	[22, 196]

 Table 2.6: Riemannian operators of (θ, α, β) -EM and BWM on $\mathcal{S}_{++}^n$.

$\mathcal{S}_{++}^n$. For (θ, α, β) -EM, the exponential map is locally defined only for tangent vectors satisfying $P^\theta + (P_\theta)_{*,P}(V) \in \mathcal{S}_{++}^n$.

- (7) **Fréchet mean.** Karcher flow [116] refers to the standard iterative solver for the Fréchet mean objective in Thm. 32.

2.9.2 Full-Rank Correlation Manifolds

Given a covariance matrix Σ , its correlation matrix is defined as

$$C = \text{Cor}(\Sigma) = \mathbb{D}(\Sigma)^{-1/2} \Sigma \mathbb{D}(\Sigma)^{-1/2}, \quad (2.97)$$

where $\mathbb{D}(\cdot)$ extracts the diagonal part of Σ as a diagonal matrix. This diagonal normalization yields a scale-invariant representation: for any positive diagonal matrix D , $\text{Cor}(D\Sigma D) = \text{Cor}(\Sigma)$. Hence, correlation matrices remove marginal scales and empha-

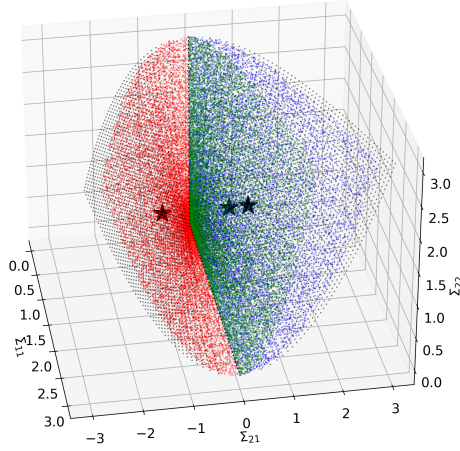


Figure 2.1: The black stars denote 2×2 correlation matrices, while the red, green, and blue dots denote corresponding SPD matrices. The black dots denote the boundary of the SPD cone.

size pairwise dependencies rather than raw variances.

Only recently have Riemannian structures been developed for correlation matrices. The space of $n \times n$ full-rank correlation matrices, denoted by $\text{Cor}^+(n)$, forms a Riemannian manifold and can be identified as a quotient manifold of the SPD manifold [62, Thm. 1]. As illustrated in Fig. 2.1, each correlation matrix corresponds to a surface in the SPD manifold. However, this quotient geometry does not guarantee uniqueness or closed forms of the Riemannian logarithm and Fréchet mean [195, Sec. 1.1]. To address this limitation, recent advances introduced five convenient Riemannian metrics on $\text{Cor}^+(n)$: the *Euclidean–Cholesky metric (ECM)* [195], *log–Euclidean–Cholesky metric (LECM)* [195], *poly–hyperbolic–Cholesky metric (PHCM)* [195], *off–log metric (OLM)* [191], and *log–scaled metric (LSM)* [191]. These metrics are pullback metrics from simpler prototype spaces: ECM, LECM, OLM, and LSM are induced from Euclidean spaces, while PHCM is induced from a product of hyperbolic open hemispheres.

We first review the associated prototype spaces.

- (1) $\text{LT}^1(n)$ is the affine space of $n \times n$ lower triangular matrices with unit diagonal.
- (2) $\text{LT}^0(n)$ is the Euclidean space of $n \times n$ lower triangular matrices with null diagonal.
- (3) $\mathcal{L}^n$ is the manifold of $n \times n$ lower triangular matrices with positive diagonals and unit row ℓ_2 -norm.
- (4) $\text{Hol}(n)$ is the Euclidean space of $n \times n$ symmetric matrices with null diagonals. The

tangent space $T_C \text{Cor}^+(n)$ at $C \in \text{Cor}^+(n)$ can be identified with $\text{Hol}(n)$.

(5) $\text{Row}_0(n)$ is the Euclidean space of $n \times n$ symmetric matrices with null row sum.

ECM. It is derived from $\text{LT}^1(n)$ by

$$\text{Cor}^+(n) \xrightleftharpoons[\Theta^{-1} = \text{Cor} \circ \text{Chol}^{-1}]{\Theta = \mathbb{D}(\text{Chol}(\cdot))^{-1} \text{Chol}(\cdot)} \text{LT}^1(n), \quad (2.98)$$

where $\Theta(C) = \mathbb{D}(\text{Chol}(C))^{-1} \text{Chol}(C)$ for any $C \in \text{Cor}^+(n)$. Here, $\text{Chol}(C)$ is the Cholesky decomposition $C = \text{Chol}(C) \text{Chol}(C)^\top$ and $\mathbb{D}(\cdot)$ returns a diagonal matrix consisting of the input diagonals. As $\text{LT}^1(n) = I_n + \text{LT}^0(n)$, ECM is essentially induced from the Euclidean space of $\text{LT}^0(n)$.

Proposition 58 (ECM). *Let $\phi^{\text{EC}}(C) = [\Theta(C)]$, where $[\cdot]$ returns a strictly lower triangular matrix. ECM over $\text{Cor}^+(n)$ is the pullback metric from the Euclidean space $\text{LT}^0(n)$ by ϕ^{EC} .*

LECM. It is defined by further pulling back ECM:

$$\text{Cor}^+(n) \xrightleftharpoons[(\log \circ \Theta)^{-1} = \text{Cor} \circ \text{Chol}^{-1} \circ \exp]{\log \circ \Theta} \text{LT}^0(n), \quad (2.99)$$

where $\log(\cdot) : \text{LT}^1(n) \rightarrow \text{LT}^0(n)$ is the matrix logarithm with the matrix exponential $\exp(\cdot)$ as its inverse.

OLM. It is derived from a permutation-invariant inner product over $\text{Hol}(n)$ by

$$\text{Cor}^+(n) \xrightleftharpoons[\text{Exp}^\circ]{\text{Log}^\circ = \text{off} \circ \log} \text{Hol}(n). \quad (2.100)$$

For any symmetric hollow matrix $H \in \text{Hol}(n)$, the operator $\mathcal{D}(H)$ returns a unique diagonal matrix such that $\text{Exp}^\circ(\cdot) : \text{Hol}(n) \ni H \mapsto \exp(\mathcal{D}(H) + H) \in \text{Cor}^+(n)$ is a diffeomorphism. As shown by Archakov and Hansen [6, Cor. 1 and Sec. 5], $\mathcal{D}(H)$ can be computed by the following exponentially converging algorithm: $D_{k+1} = D_k - \log(\mathbb{D}(\exp(D_k + H)))$, with $D_0 = \mathbf{0}_{n \times n}$ as the zero matrix.

LSM. It is derived from a permutation-invariant inner product over $\text{Row}_0(n)$ by

$$\text{Cor}^+(n) \xrightleftharpoons[\text{Exp}^\star = \text{Cor} \circ \exp]{\text{Log}^\star} \text{Row}_0(n). \quad (2.101)$$

For any correlation matrix $C \in \text{Cor}^+(n)$, there exists a unique positive diagonal matrix $\mathcal{D}^\star(C)$ such that $\text{Log}^\star(\cdot) : \text{Cor}^+(n) \ni C \mapsto \log(\mathcal{D}^\star(C)C\mathcal{D}^\star(C)) \in \text{Row}_0(n)$ is a diffeo-

morphism. As shown by Thanwerdas [191, Sec. 3.5], $\mathcal{D}^*(C)$ corresponds to the unique zero of $f : x \in \mathbb{R}_{++}^n \mapsto Cx - \frac{1}{x}$, where $\mathbb{R}_{++}^n$ denotes the set of n -dimensional positive vectors and $\frac{1}{x} = \left(\frac{1}{x_1}, \dots, \frac{1}{x_n}\right)$. This equation can be solved by damped Newton's method.

Tab. 2.7 summarizes the associated prototype spaces and diffeomorphisms. Tabs. 2.8 and 2.9 summarize the vector operations and Riemannian operators for ECM, LECM, OLM, and LSM. We use the following notation and record the following remarks.

- (1) **General notation.** Let $C, C' \in \text{Cor}^+(n)$ be correlation matrices, let $\{C_i\}_{i=1}^N \subset \text{Cor}^+(n)$, and let $V, W \in T_C \text{Cor}^+(n) \cong \text{Hol}(n)$ be tangent vectors. Let $L = \text{Chol}(C)$.
- (2) **ECM and LECM.** For any $K \in \text{LT}^1(n)$ and $X, \xi \in \text{LT}^0(n)$, the maps and differentials involved in ECM and LECM are

$$\Theta(C) = \mathbb{D}(L)^{-1}L, \quad (2.102)$$

$$\Theta^{-1}(K) = \mathbb{D}(KK^\top)^{-\frac{1}{2}}KK^\top\mathbb{D}(KK^\top)^{-\frac{1}{2}}, \quad (2.103)$$

$$\log(K) = \sum_{k=1}^{n-1} \frac{(-1)^{k-1}}{k} (K - I_n)^k, \quad (2.104)$$

$$\exp(\xi) = \sum_{k=0}^{n-1} \frac{1}{k!} \xi^k, \quad (2.105)$$

$$\Theta_{*,C}(V) = \Theta(C) (L^{-1}VL^{-\top})_{\frac{1}{2}} - \frac{1}{2}\mathbb{D}(L^{-1}VL^{-\top})\Theta(C), \quad (2.106)$$

$$(\Theta_{*,C})^{-1}(\xi) = (L\xi^\top - C\mathbb{D}(L\xi^\top))\mathbb{D}(L) + \mathbb{D}(L)(\xi L^\top - \mathbb{D}(L\xi^\top)C), \quad (2.107)$$

$$\log_{*,K}(\xi) = \sum_{k=1}^{n-1} \frac{(-1)^{k-1}}{k} \left[(K - I_n)^{k-1}\xi + \dots + \xi(K - I_n)^{k-1} \right], \quad (2.108)$$

$$\exp_{*,X}(\xi) = \sum_{k=1}^{n-1} \frac{1}{k!} (X^{k-1}\xi + X^{k-2}\xi X + \dots + \xi X^{k-1}), \quad (2.109)$$

$$(\log \circ \Theta)_{*,C}(V) = \log_{*,\Theta(C)}(\Theta_{*,C}(V)). \quad (2.110)$$

Due to the nilpotency of $\text{LT}^0(n)$, the matrix logarithm over $\text{LT}^1(n)$ and exponentiation over $\text{LT}^0(n)$ are free from eigendecomposition. Although the Euclidean inner product in the ECM and LECM columns can be any inner product, we use the canonical one in this thesis.

- (3) **OLM and LSM.** Let $H, W \in \text{Hol}(n)$, $S = H + \mathcal{D}(H)$, $\Sigma = \mathcal{D}^*(C)C\mathcal{D}^*(C)$, $X = \text{Log}^*(C) = \log(\Sigma) \in \text{Row}_0(n)$, and $Y \in \text{Row}_0(n)$. The involved maps and

differentials are [191, Thms. 2.4 and 4.1]:

$$\text{Log}_{*,C}^\circ(V) = \text{off}(\log_{*,C}(V)), \quad (2.111)$$

$$\text{Exp}_{*,H}^\circ(W) = \exp_{*,S}(W + \mathcal{D}_{*,H}(W)), \quad (2.112)$$

$$\mathcal{D}_{*,H}(W) = -\text{diag}\left((H^0)^{-1} \mathbb{D}(\exp_{*,S}(W)) \mathbf{1}\right), \quad (2.113)$$

$$H^0 = [H_{il}^0] \in \mathcal{S}_{++}^n, \quad H_{il}^0 = \sum_{j,k} U_{ij} U_{ik} U_{lj} U_{lk} [L_{\text{exp}}]_{j,k}, \quad (2.114)$$

$$\text{Log}_{*,C}^*(V) = \log_{*,\Sigma} \left(\Delta V \Delta + \frac{1}{2} (V^0 \Sigma + \Sigma V^0) \right), \quad (2.115)$$

$$\text{Exp}_{*,X}^*(Y) = \Delta^{-1} \left[\exp_{*,X}(Y) - \frac{1}{2} (\Delta^{-2} \mathbb{D}(\exp_{*,X}(Y)) \Sigma \right. \quad (2.116)$$

$$\left. + \Sigma \mathbb{D}(\exp_{*,X}(Y)) \Delta^{-2} \right] \Delta^{-1}, \quad (2.117)$$

where $S = U \text{diag}(\lambda_1, \dots, \lambda_n) U^\top$, L_{exp} is the Loewner matrix of $\exp_{*,S}$, and $\mathbf{1}$ is the vector of all ones. Here, $\log_*$ and $\exp_*$ are given by the Daleckii–Krein formula in Eqs. (2.90) and (2.91), while $\text{diag}(\cdot) : \mathbb{R}^n \rightarrow \text{Diag}(n)$ returns a diagonal matrix from an input vector. The remaining auxiliary quantities are

$$\Delta = \mathbb{D}(\Sigma)^{\frac{1}{2}}, \quad V^0 = -2 \text{diag}((I_n + \Sigma)^{-1} \Delta V \Delta \mathbf{1}). \quad (2.118)$$

- (4) **Permutation.** Let $\mathfrak{S}^n$ be the group of permutation matrices $P_\sigma = [\delta_{i,\sigma(j)}]_{1 \leq i,j \leq n}$ associated with permutations σ , and let $\mathcal{D}^\pm(n) = \{\text{diag}(\varepsilon_1, \dots, \varepsilon_n) \mid \varepsilon \in \{-1, 1\}^n\}$ be the group of diagonal matrices with entries in $\{-1, 1\}$. Thanwerdas [191, Thm. 1.1] showed that the largest congruence action on full-rank correlation matrices is the action of signed permutation matrices:

$$\begin{aligned} \star : (A, C) \in \mathfrak{S}^\pm(n) \times \text{Cor}^+(n) &\longmapsto A C A^\top \in \text{Cor}^+(n), \\ \mathfrak{S}^\pm(n) &= \mathcal{D}^\pm(n) \mathfrak{S}^n. \end{aligned} \quad (2.119)$$

As both Log_*^* and Log_*° are permutation-equivariant [191, Thms. 2.2(1) and 3.6(1)], permutation-invariant metrics over the correlation manifold can be induced by permutation-invariant inner products over $\text{Hol}(n)$ and $\text{Row}_0(n)$, respectively.

- (5) **Invariant inner products on $\text{Hol}(n)$.** For $n \geq 4$, permutation-invariant inner

products on $\text{Hol}(n)$ are [190, Thm. 8.7]

$$\begin{aligned} \langle X_1, X_2 \rangle^{(\alpha, \beta, \gamma)} &= \alpha \text{tr}(X_1 X_2) + \beta \text{Sum}(X_1 X_2) \\ &\quad + \gamma \text{Sum}(X_1) \text{Sum}(X_2), \quad \forall X_1, X_2 \in \text{Hol}(n), \end{aligned} \quad (2.120)$$

with $\alpha > 0$, $2\alpha + (n-2)\beta > 0$, and $\alpha + (n-1)(\beta + n\gamma) > 0$. For $n = 3$, permutation-invariant inner products have the same form with $\alpha = 0$:

$$\begin{aligned} \langle X_1, X_2 \rangle^{(\alpha, \beta, \gamma)} &= \beta \text{Sum}(X_1 X_2) + \gamma \text{Sum}(X_1) \text{Sum}(X_2), \\ &\quad \text{with } \beta > 0 \text{ and } \beta + 3\gamma > 0. \end{aligned} \quad (2.121)$$

For $n = 2$, they have the same form with $\alpha = \beta = 0$:

$$\langle X_1, X_2 \rangle^{(\alpha, \beta, \gamma)} = \gamma \text{Sum}(X_1) \text{Sum}(X_2), \quad \text{with } \gamma > 0. \quad (2.122)$$

- (6) **Invariant inner products on $\text{Row}_0(n)$.** For $n \geq 4$, permutation-invariant inner products on $\text{Row}_0(n)$ are [191, Thm. 4.2]

$$\begin{aligned} \langle Y_1, Y_2 \rangle^{(\alpha, \delta, \zeta)} &= \alpha \text{tr}(Y_1 Y_2) + \delta \text{tr}(\mathbb{D}(Y_1) \mathbb{D}(Y_2)) \\ &\quad + \zeta \text{tr}(Y_1) \text{tr}(Y_2), \quad \forall Y_1, Y_2 \in \text{Row}_0(n), \end{aligned} \quad (2.123)$$

with $\alpha > 0$, $n\alpha + (n-2)\delta > 0$, and $n\alpha + (n-1)(\delta + n\zeta) > 0$. For $n = 3$, the permutation-invariant inner products have the same form with $\alpha = 0$. For $n = 2$, they have the same form with $\alpha = \delta = 0$.

- (7) **OLM and LSM invariance.** Combining the permutation-equivariant diffeomorphisms with the above invariant inner products gives permutation-invariant OLM and LSM. As shown by Thanwerdas [191, Thm. 2.7], OLM is further invariant under signed permutations when $\beta = \gamma = 0$, in which case the associated $\langle \cdot, \cdot \rangle^{(\alpha, 0, 0)}$ reduces to the scaled canonical Euclidean inner product:

$$\langle V, W \rangle^{(\alpha, 0, 0)} = \alpha \langle V, W \rangle, \quad \forall V, W \in \text{Hol}(n). \quad (2.124)$$

In this thesis, we assume that $\langle \cdot, \cdot \rangle^{(\alpha, \beta, \gamma)}$ and $\langle \cdot, \cdot \rangle^{(\alpha, \delta, \zeta)}$ are the canonical Euclidean inner products. For $n \leq 3$, these inner products remain permutation invariant; the dimension-specific forms above account for redundancies among the displayed trace and sum terms.

- (8) **Inverse consistency.** We briefly review inverse-consistency, a property exclusive to LSM. The cor-inversion is defined as $\mathcal{I} : \text{Cor}^+(n) \ni C \mapsto \text{Cor}(C^{-1}) \in \text{Cor}^+(n)$ [191, Def. 1.4]. It corresponds to the matrix inversion $\text{inv} : \mathcal{S}_{++}^n \ni \Sigma \mapsto \Sigma^{-1} \in \mathcal{S}_{++}^n$, as represented by the following commuting diagram:

$$\begin{array}{ccc} \mathcal{S}_{++}^n & \xrightarrow{\text{inv}} & \mathcal{S}_{++}^n \\ \text{Cor} \downarrow & & \downarrow \text{Cor} \\ \text{Cor}^+(n) & \xrightarrow{\mathcal{I}} & \text{Cor}^+(n) \end{array} \quad (2.125)$$

As shown by Thanwerdas [191, Thm. 1.7], LSM enjoys inverse-consistency:

$$\text{Log}^*(\mathcal{I}(C)) = -\text{Log}^*(C), \quad \forall C \in \text{Cor}^+(n). \quad (2.126)$$

- (9) **Vector structure.** ECM, LECM, OLM, and LSM are all pulled back from Euclidean vector spaces. It is therefore natural to inherit vector addition and scalar multiplication from their prototype spaces. For each corresponding isometry ϕ , these operations take the form

$$C \oplus C' = \phi^{-1}(\phi(C) + \phi(C')), \quad t \odot C = \phi^{-1}(t\phi(C)), \quad (2.127)$$

with $t \in \mathbb{R}$.

2.9. Example Manifolds

Metric	Prototype space	Diffeomorphisms	Properties
ECM [195]	$\text{LT}^1(n) = \text{LT}^0(n) + I_n$	$\Theta : C \in \text{Cor}^+(n) \mapsto \mathbb{D}(\text{Chol}(C))^{-1} \text{Chol}(C) \in \text{LT}^1(n)$ $\Theta^{-1} = \text{Cor} \circ \text{Chol}^{-1} : \text{LT}^1(n) \rightarrow \text{Cor}^+(n)$	Null curvature
LECM [195]	$\text{LT}^0(n)$	$\log \circ \Theta : \text{Cor}^+(n) \rightarrow \text{LT}^0(n)$ $(\log \circ \Theta)^{-1} = \text{Cor} \circ \text{Chol}^{-1} \circ \exp : \text{LT}^0(n) \rightarrow \text{Cor}^+(n)$	Null curvature
OLM [191]	$\text{Hol}(n)$	$\text{Log}^\circ : C \in \text{Cor}^+(n) \mapsto (\text{off} \circ \log)(C) \in \text{Hol}(n)$ $(\text{Log}^\circ)^{-1} = \text{Exp}^\circ : H \in \text{Hol}(n) \mapsto \exp(\mathcal{D}(H) + H) \in \text{Cor}^+(n)$	Permutation-invariance Null curvature
LSM [191]	$\text{Row}_0(n)$	$\text{Log}^* : C \in \text{Cor}^+(n) \mapsto \log(\mathcal{D}^*(C)C\mathcal{D}^*(C)) \in \text{Row}_0(n)$ $(\text{Log}^*)^{-1} = \text{Exp}^* : R \in \text{Row}_0(n) \mapsto \text{Cor}(\exp(R)) \in \text{Cor}^+(n)$	Permutation-invariance Inverse-consistency Null curvature
PHCM [195]	$\mathbb{P}\text{HS}^{n-1}$	$\text{Chol} : \text{Cor}^+(n) \rightarrow \mathcal{L}^n \cong \mathbb{P}\text{HS}^{n-1}$ $\text{Chol}^{-1} : \mathcal{L}^n \cong \mathbb{P}\text{HS}^{n-1} \rightarrow \text{Cor}^+(n)$	Nonpositive sectional curvature

Table 2.7: Isometric prototype spaces and diffeomorphisms on the correlation manifold.

Operation	ECM	LECM
$C \oplus C'$	$(\phi^{\text{EC}})^{-1} (\phi^{\text{EC}}(C) + \phi^{\text{EC}}(C'))$	$(\log \circ \Theta)^{-1} (\log \circ \Theta(C) + \log \circ \Theta(C'))$
$t \odot C$	$(\phi^{\text{EC}})^{-1} (t\phi^{\text{EC}}(C))$	$(\log \circ \Theta)^{-1} (t \log \circ \Theta(C))$
$g_C(V, W)$	$\langle \Theta_{*,C}(V), \Theta_{*,C}(W) \rangle$	$\langle (\log \circ \Theta)_{*,C}(V), (\log \circ \Theta)_{*,C}(W) \rangle$
$\text{Exp}_C(V)$	$\Theta^{-1} (\Theta(C) + \Theta_{*,C}(V))$	$(\log \circ \Theta)^{-1} (\log \circ \Theta(C) + (\log \circ \Theta)_{*,C}(V))$
$\text{Log}_C(C')$	$\Theta_{*,\Theta(C)}^{-1} (\Theta(C') - \Theta(C))$	$(\log \circ \Theta)_{*,\log \circ \Theta(C)}^{-1} (\log \circ \Theta(C') - \log \circ \Theta(C))$
$\gamma(t; C, C')$	$\Theta^{-1} ((1-t)\Theta(C) + t\Theta(C'))$	$(\log \circ \Theta)^{-1} ((1-t)\log \circ \Theta(C) + t \log \circ \Theta(C'))$
$d(C, C')$	$\ \Theta(C) - \Theta(C')\ $	$\ \log \circ \Theta(C) - \log \circ \Theta(C')\ $
Fréchet mean	$\Theta^{-1} \left(\frac{1}{N} \sum_{i=1}^N \Theta(C_i) \right)$	$(\log \circ \Theta)^{-1} \left(\frac{1}{N} \sum_{i=1}^N (\log \circ \Theta)(C_i) \right)$
Curvature	0	0
$\text{PT}_{C \rightarrow C'}(V)$	$(\Theta_{*,C'})^{-1} (\Theta_{*,C}(V))$	$((\log \circ \Theta)_{*,C'})^{-1} ((\log \circ \Theta)_{*,C}(V))$

Table 2.8: Vector operations and Riemannian operators under ECM and LECM.

Operation	OLM	LSM
$C \oplus C'$	$\text{Exp}^\circ (\text{Log}^\circ(C) + \text{Log}^\circ(C'))$	$\text{Exp}^* (\text{Log}^*(C) + \text{Log}^*(C'))$
$t \odot C$	$\text{Exp}^\circ (t \text{Log}^\circ(C))$	$\text{Exp}^* (t \text{Log}^*(C))$
$g_C(V, W)$	$\langle \text{Log}_{*,C}^\circ(V), \text{Log}_{*,C}^\circ(W) \rangle^{(\alpha, \beta, \gamma)}$	$\langle \text{Log}_{*,C}^*(V), \text{Log}_{*,C}^*(W) \rangle^{(\alpha, \delta, \zeta)}$
$\text{Exp}_C(V)$	$\text{Exp}^\circ (\text{Log}^\circ(C) + \text{Log}_{*,C}^\circ(V))$	$\text{Exp}^* (\text{Log}^*(C) + \text{Log}_{*,C}^*(V))$
$\text{Log}_C(C')$	$\text{Exp}_{*,\text{Log}^\circ(C)}^\circ (\text{Log}^\circ(C') - \text{Log}^\circ(C))$	$\text{Exp}_{*,\text{Log}^*(C)}^* (\text{Log}^*(C') - \text{Log}^*(C))$
$\gamma(t; C, C')$	$\text{Exp}^\circ ((1-t)\text{Log}^\circ(C) + t\text{Log}^\circ(C'))$	$\text{Exp}^* ((1-t)\text{Log}^*(C) + t\text{Log}^*(C'))$
$d(C, C')$	$\ \text{Log}^\circ(C) - \text{Log}^\circ(C')\ ^{(\alpha, \beta, \gamma)}$	$\ \text{Log}^*(C) - \text{Log}^*(C')\ ^{(\alpha, \delta, \zeta)}$
Fréchet mean	$\text{Exp}^\circ \left(\frac{1}{N} \sum_{i=1}^N \text{Log}^\circ(C_i) \right)$	$\text{Exp}^* \left(\frac{1}{N} \sum_{i=1}^N \text{Log}^*(C_i) \right)$
Curvature	0	0
$\text{PT}_{C \rightarrow C'}(V)$	$(\text{Log}_{*,C'}^\circ)^{-1} (\text{Log}_{*,C}^\circ(V))$	$(\text{Log}_{*,C'}^*)^{-1} (\text{Log}_{*,C}^*(V))$

Table 2.9: Vector operations and Riemannian operators under OLM and LSM.

PHCM [195, Def. 4.3 and Thm. 4.4]. It is defined through the Cholesky decomposition and a product of open-hemisphere models of hyperbolic space. For a correlation matrix $C \in \text{Cor}^+(n)$, let $L = \text{Chol}(C)$. For $k = 2, \dots, n$, define the nonzero part of the k -th row of L by

$$\ell_k = (L_{k1}, \dots, L_{kk}) \in \text{HS}^{k-1}, \quad \text{HS}^{k-1} = \{x \in \mathbb{R}^k \mid \|x\| = 1, x_k > 0\}. \quad (2.128)$$

Thus, $\mathcal{L}^n$ is identified with the product of $n-1$ open hemispheres, denoted by $\mathbb{PHS}^{n-1} = \prod_{i=1}^{n-1} \text{HS}^i$. The first row, for which $L_{11} = 1$ and $\text{HS}^0 = \{1\}$, is trivial and omitted from the product. PHCM is the pullback by the Cholesky decomposition of the product metric on $\prod_{i=1}^{n-1} (\text{HS}^i, \alpha_i g^{\text{HS}^i})$, where each α_i is a positive weight and g^{HS^i} denotes the metric tensor on HS^i . In particular, PHCM with all weights equal to 1 is called the canonical PHCM, on which we focus below. Given $C \in \text{Cor}^+(n)$ and $L = \text{Chol}(C) \in \mathcal{L}^n$, define

$$\Psi = \psi^1 \times \dots \times \psi^{n-1} : \mathcal{L}^n \rightarrow \prod_{i=1}^{n-1} \text{HS}^i, \quad (2.129)$$

where

$$\psi^i(L) = (L_{i+1,1}, \dots, L_{i+1,i+1}) \in \text{HS}^i. \quad (2.130)$$

For $Z \in T_L \mathcal{L}^n$, its differential is the corresponding row extraction,

$$\psi_{*,L}^i(Z) = (Z_{i+1,1}, \dots, Z_{i+1,i+1}) \in T_{\psi^i(L)} \text{HS}^i, \quad \Psi_{*,L} = \psi_{*,L}^1 \times \dots \times \psi_{*,L}^{n-1}. \quad (2.131)$$

The Riemannian operators under PHCM are obtained from the product geometry and the geometry of each HS^i . Let $C' \in \text{Cor}^+(n)$, $L' = \text{Chol}(C')$, and $V, W \in T_C \text{Cor}^+(n)$. Write $a_i = \psi^i(L)$, $a'_i = \psi^i(L')$, and $\xi_i(V) = \psi_{*,L}^i(\text{Chol}_{*,C}(V))$. Then

$$g_C(V, W) = \left\langle \mathbb{D}(L)^{-1} L (L^{-1} V L^{-\top})_{\frac{1}{2}}, \mathbb{D}(L)^{-1} L (L^{-1} W L^{-\top})_{\frac{1}{2}} \right\rangle, \quad (2.132)$$

$$\text{Exp}_C(V) = \text{Chol}^{-1} \left(\Psi^{-1} \left(\text{Exp}_{a_1}^{\text{HS}^1}(\xi_1(V)), \dots, \text{Exp}_{a_{n-1}}^{\text{HS}^{n-1}}(\xi_{n-1}(V)) \right) \right), \quad (2.133)$$

$$\text{Log}_C(C') = \text{Chol}_{*,L}^{-1} \left((\Psi_{*,L})^{-1} \left(\text{Log}_{a_1}^{\text{HS}^1}(a'_1), \dots, \text{Log}_{a_{n-1}}^{\text{HS}^{n-1}}(a'_{n-1}) \right) \right), \quad (2.134)$$

$$\gamma(t; C, C') = \text{Chol}^{-1} \left(\Psi^{-1} \left(\gamma^{\text{HS}^1}(t; a_1, a'_1), \dots, \gamma^{\text{HS}^{n-1}}(t; a_{n-1}, a'_{n-1}) \right) \right), \quad (2.135)$$

$$d(C, C')^2 = \sum_{i=1}^{n-1} \operatorname{arccosh} \left(1 + \frac{1 - \langle a_i, a'_i \rangle}{L_{i+1, i+1} L'_{i+1, i+1}} \right)^2. \quad (2.136)$$

Here, $\operatorname{Log}^{\operatorname{HS}^i}$, $\operatorname{Exp}^{\operatorname{HS}^i}$, and $\gamma^{\operatorname{HS}^i}$ are the corresponding operators on HS^i ; the distance and these closed forms follow from the hyperboloid–hemisphere isometry [195, Thm. 4.2].

2.9.3 Grassmannian Manifolds

The Grassmannian has been widely applied in machine learning, ranging from action recognition [108] to question answering [159], shape generation [224], image classification [204], and signal analysis [210].

The *Grassmannian manifold* is the set of p -dimensional subspaces of $\mathbb{R}^n$ [19]. It has two matrix representations: the *projector perspective (PP)* and the *orthonormal-basis perspective (ONB)*:

$$\begin{aligned} \widetilde{\operatorname{Gr}}(p, n) &= \{P \in \mathcal{S}^n \mid P^2 = P, \quad \operatorname{rank}(P) = p\}, \\ \operatorname{Gr}(p, n) &= \left\{ [U] \mid [U] := \left\{ \tilde{U} \in \operatorname{St}(p, n) \mid \tilde{U} = UR, \quad R \in \operatorname{O}(p) \right\} \right\}, \end{aligned} \quad (2.137)$$

where $\mathcal{S}^n$ is the Euclidean space of symmetric matrices, $\operatorname{St}(p, n)$ is the Stiefel manifold, and $\operatorname{O}(p)$ is the orthogonal group. By abuse of notation, we use $[U]$ and U interchangeably for elements of $\operatorname{Gr}(p, n)$, where the $n \times p$ column-wise orthonormal matrix U is taken as a representative of an equivalence class. Helmke and Moore [100] show that the ONB perspective is diffeomorphic to the PP representation by

$$\pi : \operatorname{Gr}(p, n) \ni U \mapsto UU^\top \in \widetilde{\operatorname{Gr}}(p, n). \quad (2.138)$$

As shown by Nguyen [157, Sec. 3.2] and Nguyen and Yang [159, Sec. 2.3.1], the Grassmannian admits gyro-structures defined by Eqs. (2.77) to (2.83).

Under the ONB perspective, tangent vectors are represented by horizontal lifts. Given an orthogonal complement $U_\perp \in \operatorname{St}(n-p, n)$ of U , every tangent vector $\Delta \in T_U \operatorname{Gr}(p, n)$ can be written as

$$\Delta = U_\perp B, \quad B \in \mathbb{R}^{(n-p) \times p}. \quad (2.139)$$

Under the PP perspective, every point can be written as $P = O \tilde{I}_{p, n} O^\top$ with $O \in \operatorname{O}(n)$,

Operator	ONB: $\text{Gr}(p, n)$	PP: $\widetilde{\text{Gr}}(p, n)$
$g_U(\Delta, \Xi)$ or $\widetilde{g}_P(A_1, A_2)$	$g_U(\Delta, \Xi) = \langle \Delta, \Xi \rangle$	$\widetilde{g}_P(A_1, A_2) = \frac{1}{2} \langle A_1, A_2 \rangle$
$d(U, V)$ or $\widetilde{d}(P, Q)$	$\ \arccos(\Sigma)\ $ $U^\top V \stackrel{\text{SVD}}{:=} O \Sigma R^\top$	$\frac{1}{2\sqrt{2}} \ \log((I_n - 2Q)(I_n - 2P))\ $
$\text{Log}_U V$ or $\widetilde{\text{Log}}_P Q$	$O \arctan(\Sigma) R^\top$ $(I_n - UU^\top) V (U^\top V)^{-1} \stackrel{\text{SVD}}{:=} O \Sigma R^\top$	$\frac{1}{2} [\log((I_n - 2Q)(I_n - 2P)), P]$
$\text{Exp}_U \Delta$ or $\widetilde{\text{Exp}}_P A$	$U R \cos(\Sigma) R^\top + O \sin(\Sigma) R^\top$ $\Delta \stackrel{\text{SVD}}{:=} O \Sigma R^\top$	$\exp([A, P]) P \exp(-[A, P])$
$\gamma(t; U, V)$ or $\widetilde{\gamma}(t; P, Q)$	$U R \cos(t\Sigma) R^\top + O \sin(t\Sigma) R^\top$ $\text{Log}_U V \stackrel{\text{SVD}}{:=} O \Sigma R^\top$	$\exp(t[\widetilde{\text{Log}}_P Q, P]) P \exp(-t[\widetilde{\text{Log}}_P Q, P])$
$\text{PT}_{U \rightarrow V}(\Delta)$ or $\text{PT}_{P \rightarrow Q}(A)$	$\left([UR \ O] \begin{bmatrix} -\sin(\Sigma) \\ \cos(\Sigma) \end{bmatrix} O^\top + I_n - OO^\top \right) \Delta$ $\text{Log}_U V \stackrel{\text{SVD}}{:=} O \Sigma R^\top$	$\exp([\widetilde{\text{Log}}_P Q, P]) A \exp(-[\widetilde{\text{Log}}_P Q, P])$
$\text{FM}(\{U_i\})$ or $\widetilde{\text{FM}}(\{P_i\})$	Karcher flow	Karcher flow
References	[71, 19]	[19]

Table 2.10: Riemannian operators on the Grassmannian under ONB and PP.

Operator	ONB: $\text{Gr}(p, n)$	PP: $\widetilde{\text{Gr}}(p, n)$
Gyroaddition	$U \oplus^{\text{Gr}} V = \exp(\Omega_U) V$	$P \widetilde{\oplus}^{\text{Gr}} Q = \exp(\Omega_P) Q \exp(-\Omega_P)$
Gyro identity	$I_{p,n}$	$\widetilde{I}_{p,n}$
Scalar gyromultiplication	$t \odot^{\text{Gr}} U = \exp(t\Omega_U) I_{p,n}$	$t \widetilde{\odot}^{\text{Gr}} P = \exp(t\Omega_P) \widetilde{I}_{p,n} \exp(-t\Omega_P)$
Gyro-inverse	$\ominus^{\text{Gr}} U = \exp(-\Omega_U) I_{p,n}$	$\widetilde{\ominus}^{\text{Gr}} P = \exp(-\Omega_P) \widetilde{I}_{p,n} \exp(\Omega_P)$
References	[159]	[157]

Table 2.11: Gyro operators on the Grassmannian under ONB and PP.

and every tangent vector $A \in T_P \widetilde{\text{Gr}}(p, n)$ can be written as

$$A = O \begin{bmatrix} \mathbf{0} & B^\top \\ B & \mathbf{0} \end{bmatrix} O^\top, \quad B \in \mathbb{R}^{(n-p) \times p}. \quad (2.140)$$

Tab. 2.10 and Tab. 2.11 summarize the Riemannian and gyro operators, respectively; their formulas use the *singular value decomposition (SVD)*. In these tables, $U, V \in \text{Gr}(p, n)$ and $\Delta, \Xi \in T_U \text{Gr}(p, n)$ refer to the ONB perspective, while $P, Q \in \widetilde{\text{Gr}}(p, n)$ and $A, A_1, A_2 \in T_P \widetilde{\text{Gr}}(p, n)$ refer to the PP perspective. To distinguish the two perspectives, PP Riemannian operators are marked by a tilde, such as $\widetilde{\text{Log}}$ and $\widetilde{\text{Exp}}$. The differential of π is $\pi_{*,U}(\Delta) = U\Delta^\top + \Delta U^\top$. For gyro operators, define $\overline{P} = \widetilde{\text{Log}}_{\widetilde{I}_{p,n}}(P)$. Let $\Omega_U = [\overline{UU}^\top, \widetilde{I}_{p,n}]$ and $\Omega_P = [\overline{P}, \widetilde{I}_{p,n}]$. The identities are $I_{p,n} = [I_p, \mathbf{0}]^\top$ and $\widetilde{I}_{p,n} = I_{p,n} I_{p,n}^\top$.

2.9. Example Manifolds

Operator	$R \oplus S$	$g_R(A_1, A_2)$	$d(R, S)$	$\text{Log}_R S$	$\text{Exp}_R(A)$	$\gamma(t; R, S)$	FM
Expression	RS	$\langle A_1, A_2 \rangle$	$\ \log(R^\top S)\ $	$R \log(R^\top S)$	$R \exp(R^\top A)$	$R \exp(t \log(R^\top S))$	Karcher flow
References					[146]		

Table 2.12: Lie group structures and Riemannian operators on rotation matrices.

2.9.4 Special Orthogonal Groups

The set of $n \times n$ rotation matrices forms a Lie group, known as the *special orthogonal group* and denoted by $\text{SO}(n)$ [197]:

$$\text{SO}(n) = \{R \in \mathbb{R}^{n \times n} \mid R^\top R = I_n, \det(R) = 1\}. \quad (2.141)$$

Its group operation is the matrix product, with the identity matrix as the neutral element. Any tangent vector $A \in T_R \text{SO}(n)$ can be represented as $A = RV$, with $V \in \mathfrak{so}(n)$. Here, $\mathfrak{so}(n)$ is the Lie algebra of $\text{SO}(n)$, which is the tangent space at the identity matrix, formed by the set of $n \times n$ skew-symmetric matrices:

$$\mathfrak{so}(n) = \{\Omega \in \mathbb{R}^{n \times n} \mid \Omega^\top = -\Omega\}. \quad (2.142)$$

The Fréchet mean can be obtained by Karcher flow [146]. Furthermore, if all rotations lie in a closed ball of radius $r < \pi/2$, then Karcher flow converges to the unique mean [146, Thm. 5]. Given $R, S \in \text{SO}(n)$ and tangent vectors $A, A_1, A_2 \in T_R \text{SO}(n)$, Tab. 2.12 summarizes all the associated operators on $\text{SO}(n)$.

Across these matrix-manifold examples, the tables expose the Euclidean template behind the spaces used later: flat pullback metrics use ordinary addition and scaling in a chart, quotient metrics use horizontal representatives, product metrics act factor by factor, and Lie-group metrics use group translation.

2.9.5 Constant-Curvature Manifolds

Definition 59 (Constant-curvature space [165, Ch. 8]). A *constant-curvature space (CCS)* is a complete, simply connected, n -dimensional Riemannian manifold of constant curvature K .

By O’Neill [165, Cor. 8.25], any two CCSs with the same dimension and the same curvature K are isometric. Thus, a CCS is determined up to isometry by K : Euclidean space corresponds to $K = 0$, spherical space to $K > 0$, and hyperbolic space to $K < 0$. In the following, we introduce several concrete models of these spaces that are used

later: the K -stereographic model, the K -radius model, and the Beltrami–Klein model.

K -stereographic model [13]. This model has shown success in different applications, including computer vision [201], natural language processing [76, 180], graph learning [13, 85, 86], and astronomy [44]. It is defined as a model $\mathfrak{st}_K^n$ with the conformal metric

$$\langle u, v \rangle_x^{\text{st}} = (\lambda_x^K)^2 \langle u, v \rangle, \quad \lambda_x^K = \frac{2}{1 + K \|x\|^2}, \quad (2.143)$$

where $K \in \mathbb{R}$ is the constant curvature and λ_x^K is a conformal factor. In particular, $\mathfrak{st}_K^n$ is the scaled $\mathbb{R}^n$ when $K = 0$. It unifies the spherical projected hypersphere $\mathbb{D}_K^n$, Euclidean space $\mathbb{R}^n$, and the hyperbolic Poincaré ball $\mathbb{P}_K^n$:

$$\mathfrak{st}_K^n = \begin{cases} \mathbb{D}_K^n = \mathbb{R}^n, & \text{For } K > 0, \text{ spherical geometry,} \\ \mathbb{R}^n, & \text{For } K = 0, \text{ Euclidean geometry,} \\ \mathbb{P}_K^n = \{x \in \mathbb{R}^n \mid \|x\|^2 < -\frac{1}{K}\}, & \text{For } K < 0, \text{ hyperbolic geometry.} \end{cases} \quad (2.144)$$

Although $\mathbb{D}_K^n = \mathbb{R}^n$ for $K > 0$, its metric is conformal to the Euclidean one. We abbreviate the K -stereographic model as the stereographic model. Bachmann et al. [13, Eqs. 2–3] show that this model admits a gyro-structure.

K -radius model [182]. This model has been effective in various applications [38, 45, 15, 166, 99, 119]. It provides an extrinsic representation of the space with constant curvature $K \in \mathbb{R}$, encompassing the sphere $\mathbb{S}_K^n$, Euclidean space $\mathbb{R}^n$, and the Lorentz, or hyperboloid, model $\mathbb{L}_K^n$:

$$\mathcal{M}_K^n = \begin{cases} \mathbb{S}_K^n = \{x \in \mathbb{R}^{n+1} \mid \|x\|^2 = \frac{1}{K}\}, & \text{For } K > 0, \text{ spherical geometry,} \\ \mathbb{R}^n, & \text{For } K = 0, \text{ Euclidean geometry,} \\ \mathbb{L}_K^n = \{x \in \mathbb{R}^{n+1} \mid \langle x, x \rangle_{\mathcal{L}} = \frac{1}{K}, x_t > 0\}, & \text{For } K < 0, \text{ hyperbolic geometry,} \end{cases}$$

where $\langle x, x \rangle_{\mathcal{L}} = \|x_s\|^2 - x_t^2$ is the Lorentzian quadratic form. Following the conventions of the hyperboloid, we write $x = (x_t, x_s^\top)^\top$, where $x_t \in \mathbb{R}$ is the time component and $x_s \in \mathbb{R}^n$ is the spatial component [173]. When $K \neq 0$, the model can be written compactly as

$$\mathcal{M}_K^n = \left\{ x \in \mathbb{R}^{n+1} \mid \langle x, x \rangle_K = \frac{1}{K} \right\}, \quad \langle \cdot, \cdot \rangle_K = \begin{cases} \langle \cdot, \cdot \rangle, & K > 0, \\ \langle \cdot, \cdot \rangle_{\mathcal{L}}, & K < 0, \end{cases} \quad (2.145)$$

with the hyperbolic branch restricted to $x_t > 0$. We abbreviate the K -radius model

2.9. Example Manifolds

Operator	Stereographic model $\mathfrak{st}_K^n$	Beltrami–Klein model $\mathbb{K}_K^n$
Gyroaddition	$x \oplus_K y = \frac{(1 - 2K \langle x, y \rangle - K \ y\ ^2)x + (1 + K \ x\ ^2)y}{1 - 2K \langle x, y \rangle + K^2 \ x\ ^2 \ y\ ^2}$	$x \oplus_E y = \frac{1}{1 - K \langle x, y \rangle} \left(x + \frac{1}{\gamma_x^K} y - K \frac{\gamma_x^K}{1 + \gamma_x^K} \langle x, y \rangle x \right)$
Gyro identity	$\mathbf{0}$	$\mathbf{0}$
Scalar gyromultiplication	$t \odot_K x = \frac{\tan_K \left(t \tan_K^{-1} \left(\sqrt{ K } \ x\ \right) \right)}{\sqrt{ K }} \frac{x}{\ x\ }$	$t \odot_E x = \frac{\tanh \left(t \tanh^{-1} \left(\sqrt{-K} \ x\ \right) \right)}{\sqrt{-K}} \frac{x}{\ x\ }$
Gyro-inverse	$\ominus_K x = -x$	$\ominus_E x = -x$
Gyration	$\text{gyr}[x, y]z = z + \frac{2(A_{\text{st}}x + B_{\text{st}}y)}{D_{\text{st}}}$	$\text{gyr}[x, y]z = z + \frac{A_E x + B_E y}{D_E}$
References	[200]	[200]

Table 2.13: Gyro operators on stereographic and Beltrami–Klein models.

as the radius model. On its negative-curvature branch, we use $\mathcal{M}_K^n = \mathbb{L}_K^n = \mathbb{H}_K^n$ interchangeably; in hyperbolic-only contexts, the specializations of $\oplus_K^M$, $\ominus_K^M$, and $\odot_K^M$ are denoted by $\oplus_{\mathbb{L}}$, $\ominus_{\mathbb{L}}$, and $\odot_{\mathbb{L}}$, respectively. Its gyro-structure is introduced later in Sec. 3.3.

Beltrami–Klein model. There are five models of hyperbolic space [33]. Apart from the above Poincaré ball and hyperboloid models, we further study the Beltrami–Klein model:

$$\mathbb{K}_K^n = \left\{ x \in \mathbb{R}^n \mid \|x\|^2 < -\frac{1}{K} \right\}, \quad \text{with } g_x^{\mathbb{K}}(v, w) = \frac{\langle v, w \rangle}{1 + K \|x\|^2} - \frac{K \langle x, v \rangle \langle x, w \rangle}{(1 + K \|x\|^2)^2},$$

where $K < 0$ is the constant curvature and $g^{\mathbb{K}}$ is its Riemannian metric. Although the Poincaré ball and Beltrami–Klein models share the same underlying set, their Riemannian metrics differ. This model admits an Einstein gyrovector space [200, Sec. 6.18].

The gyro-structures for the stereographic and Beltrami–Klein models are summarized in Tab. 2.13. In the table, $x, y, z \in \mathfrak{st}_K^n$ for the stereographic model, $x, y, z \in \mathbb{K}_K^n$ for the Beltrami–Klein model, $t \in \mathbb{R}$, and $\gamma_x^K = (1 + K \|x\|^2)^{-1/2}$ is the Einstein gamma factor. The stereographic convention is $\tan_K = \tanh$ for $K < 0$ and $\tan_K = \tan$ for $K > 0$.

The stereographic gyration in Tab. 2.13, following Bachmann et al. [13, App. C.2.6], is

$$\text{gyr}[x, y]z = z + 2 \frac{A_{\text{st}}x + B_{\text{st}}y}{D_{\text{st}}}, \quad (2.146)$$

with

$$\begin{aligned} A_{\text{st}} &= -K^2 \langle x, z \rangle \|y\|^2 - K \langle y, z \rangle + 2K^2 \langle x, y \rangle \langle y, z \rangle, \\ B_{\text{st}} &= -K^2 \langle y, z \rangle \|x\|^2 + K \langle x, z \rangle, \\ D_{\text{st}} &= 1 - 2K \langle x, y \rangle + K^2 \|x\|^2 \|y\|^2 \\ &= (1 - K \langle x, y \rangle)^2 + K^2 (\|x\|^2 \|y\|^2 - \langle x, y \rangle^2) \geq 0. \end{aligned} \quad (2.147)$$

The Cauchy–Schwarz inequality gives the last relation. The gyration formula applies whenever $D_{\text{st}} > 0$; singular positive-curvature configurations with $D_{\text{st}} = 0$ are excluded. The Einstein gyration coefficients for the Beltrami–Klein model are

$$\begin{aligned}
 A_E &= K \frac{(\gamma_x^K)^2}{\gamma_x^K + 1} (\gamma_y^K - 1) \langle x, z \rangle - K \gamma_x^K \gamma_y^K \langle y, z \rangle \\
 &\quad + 2K^2 \frac{(\gamma_x^K)^2 (\gamma_y^K)^2}{(\gamma_x^K + 1)(\gamma_y^K + 1)} \langle x, y \rangle \langle y, z \rangle, \\
 B_E &= K \frac{\gamma_y^K}{\gamma_y^K + 1} (\gamma_x^K (\gamma_y^K + 1) \langle x, z \rangle + (\gamma_x^K - 1) \gamma_y^K \langle y, z \rangle), \\
 D_E &= 1 + \gamma_x^K \gamma_y^K (1 - K \langle x, y \rangle) = 1 + \gamma_{x \oplus_E y}^K.
 \end{aligned} \tag{2.148}$$

Tab. 2.14 summarizes the associated Riemannian operators for the stereographic and radius models when $K \neq 0$; at $K = 0$, they reduce to the usual Euclidean operators. On the sphere, logarithmic maps and parallel transports are restricted away from antipodal pairs. The gyro operators of the radius model and the closed-form Riemannian operators of the Beltrami–Klein model are introduced later in Sec. 3.3.

In Tab. 2.14, for $K \neq 0$, the curvature-aware trigonometric functions are

$$\begin{aligned}
 \tan_K(\cdot) &= \begin{cases} \tan(\cdot), & K > 0, \\ \tanh(\cdot), & K < 0, \end{cases} & \sin_K(\cdot) &= \begin{cases} \sin(\cdot), & K > 0, \\ \sinh(\cdot), & K < 0, \end{cases} \\
 \cos_K(\cdot) &= \begin{cases} \cos(\cdot), & K > 0, \\ \cosh(\cdot), & K < 0, \end{cases}
 \end{aligned}$$

All ratios in these tables are understood by continuous extension at removable zero denominators: in particular, $t \odot_K \mathbf{0} = \mathbf{0}$, $\text{Exp}_x(0) = x$, and $\text{Log}_x(x) = 0$. On the positive-curvature branch, logarithmic maps and parallel transports are restricted away from antipodal pairs and other stated singular configurations. For the radius model, if $x \in \mathcal{M}_K^n$ and $v \in T_x \mathcal{M}_K^n$, then $\|v\|_K = \sqrt{\langle v, v \rangle_K}$; for $K < 0$, the Lorentzian form is positive definite only after restriction to this tangent space. These formulas show that constant-curvature neural layers can be implemented by choosing a model, applying the matching exponential, logarithmic, and transport maps, and reducing to Euclidean vector operations when $K = 0$.

2.9. Example Manifolds

Operator	Stereographic: $\mathbb{P}_K^n, \mathbb{D}_K^n$	Radius: $\mathbb{L}_K^n, \mathbb{S}_K^n$
Metric	$\langle u, v \rangle_x = (\lambda_x^K)^2 \langle u, v \rangle$	$\langle u, v \rangle_x = \langle u, v \rangle_K$
Geodesic distance	$\frac{2}{\sqrt{ K }} \tan_K^{-1} \left(\sqrt{ K } \ -x \oplus_K y \ \right)$	$\frac{1}{\sqrt{ K }} \cos_K^{-1} (K \langle x, y \rangle_K)$
Exponential map	$x \oplus_K \left(\tan_K \left(\sqrt{ K } \frac{\lambda_x^K \ v\ }{2} \right) \frac{v}{\sqrt{ K } \ v\ } \right)$	$\cos_K \left(\sqrt{ K } \ v\ _K \right) x + \frac{\sin_K \left(\sqrt{ K } \ v\ _K \right)}{\sqrt{ K } \ v\ _K} v$
Logarithmic map	$\frac{2 \tan_K^{-1} \left(\sqrt{ K } \ -x \oplus_K y \ \right)}{\sqrt{ K } \lambda_x^K} \frac{-x \oplus_K y}{\ -x \oplus_K y \ }$	$\frac{\cos_K^{-1}(\beta)}{\sqrt{\text{sign}(K)(1-\beta^2)}} (y - \beta x), \beta = K \langle x, y \rangle_K$
Parallel transport	$\frac{\lambda_x^K}{\lambda_y^K} \text{gyr}[y, -x]v$	$v - \frac{K \langle y, v \rangle_K}{1+K \langle x, y \rangle_K} (x + y)$
Fréchet mean	$K < 0$: [142, Alg. 1] $K > 0$: Karcher flow	$K < 0$: [142, Alg. 3] $K > 0$: Karcher flow
References	[182]	[182]

Table 2.14: Riemannian operator templates for stereographic and radius constant-curvature coordinates.

Chapter 3

Riemannian Batch Normalization

3.1 Introduction

Motivated by the great success of normalization techniques [109, 12, 198, 219], researchers have sought to devise normalization layers tailored for manifold-valued data. Brooks et al. [31] introduced *Riemannian Batch Normalization (RBN)* designed specifically for the SPD manifold, with the ability to normalize the Riemannian mean. Kobler et al. [123] extended this approach to further control the Riemannian variance. *However, the above methods are constrained to AIM on the SPD manifold, limiting their applicability.* On the other hand, Chakraborty [34] proposed two distinct Riemannian normalization frameworks: one for Riemannian homogeneous spaces [34, Algs. 1–2] and another for matrix Lie groups [34, Algs. 3–4]. *Nonetheless, the normalization designed for Riemannian homogeneous spaces can normalize neither the mean nor the variance, while the one for matrix Lie groups is confined to a specific type of distance [34, Sec. 3.2].* Meanwhile, Lou et al. [142, Alg. 2] proposed an RBN layer for general geometries. However, similar to Chakraborty [34, Algs. 1–2], it lacks theoretical guarantees for normalizing sample statistics. Therefore, a principled Riemannian normalization framework capable of controlling both Riemannian mean and variance remains unexplored.

Given that *Batch Normalization (BN)* [109] serves as the foundational prototype for various types of normalization, this chapter focuses on RBN, with the potential to be extended to other normalization variants. We first present a general RBN framework for Lie groups, referred to as *Lie Group Batch Normalization (LieBN)*, which can normalize both the Riemannian mean and variance under invariant metrics. To extend this normalization principle beyond Lie groups, we next introduce pseudo-reductive gyrogroups,

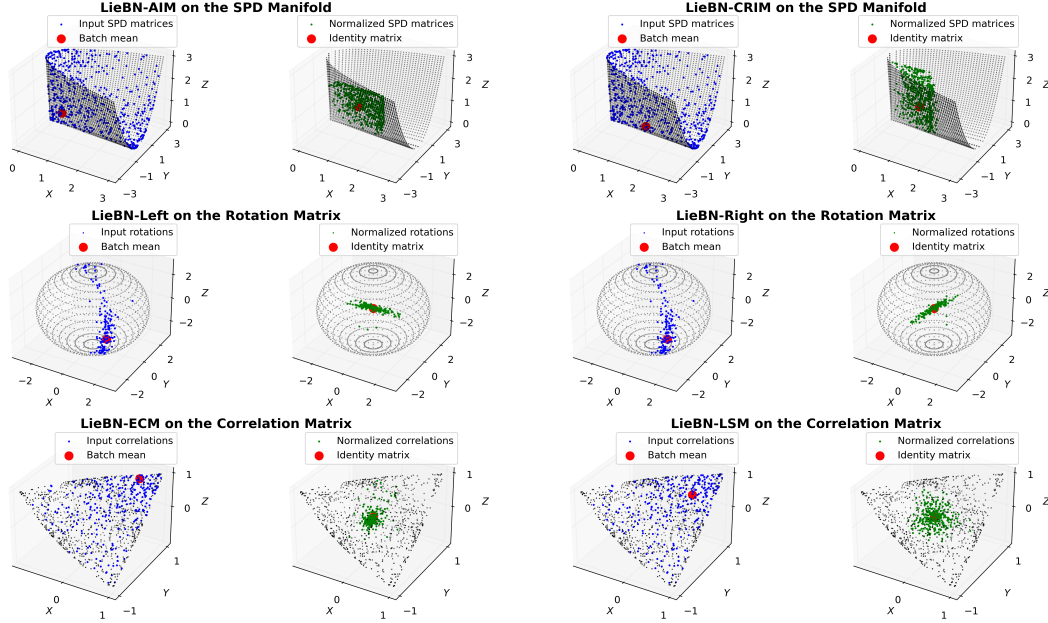


Figure 3.1: Illustration of LieBN on the SPD, rotation, and correlation Lie groups. The 2×2 SPD, 3×3 rotation, and 3×3 correlation manifolds can be embedded into $\mathbb{R}^3$ as an open cone [220], a closed ball with antipodal points identified [95], and an open ellptope [195], respectively. LieBN is illustrated by (1) the left-invariant AIM and the proposed right-invariant CRIM geometry on the SPD manifold, (2) left or right translation under a bi-invariant metric on the rotation manifold, and (3) the bi-invariant ECM and LSM geometry on the correlation manifold. On the SPD and correlation manifolds, the batch mean and variance of the same input samples differ under different geometries. In all sub-figures, the black, blue, green, and red dots denote the boundary of the space, the input Lie group samples, the normalized samples, and the batch mean, respectively. As illustrated, our LieBN effectively normalizes the Lie group distribution.

a relaxation of classical gyrogroups that also encompasses Lie groups as special cases. Building on this algebraic structure, we develop *Gyrogroup Batch Normalization* (*GyroBN*), which generalizes LieBN beyond group structures. We then instantiate LieBN and GyroBN on SPD manifolds, rotation matrices, correlation matrices, the Grassmannian, and different CCSs. Extensive experiments across multiple tasks demonstrate the effectiveness of this unified design.

3.2 Lie Group Batch Normalization

3.2.1 Introduction

Since several manifold-valued measurements form Lie groups, such as SPD manifolds [9, 137, 195], special orthogonal groups $SO(n)$ [28], and full-rank correlation matrices

[195, 191], we direct our attention to Lie groups. As each Lie group naturally admits left- and right-invariant metrics [69, Ch. 1.2], we propose a principled framework for RBN over Lie groups under invariant metrics, referred to as LieBN. Compared to previous work, our framework provides a theoretical guarantee for normalizing the Riemannian sample mean and variance.

Empirically, we focus on the SPD, special orthogonal, and full-rank correlation manifolds. On SPD manifolds, we generalize three existing Lie group structures into parameterized ones by matrix power deformation. Additionally, we propose a novel right-invariant metric, which, to the best of our knowledge, is the **first** non-trivial right-invariant SPD metric¹, referred to as the *Cholesky Right Invariant Metric (CRIM)*. We then instantiate our LieBN framework on SPD manifolds under these four Lie group structures. For rotation matrices, we adopt the popular bi-invariant metric [28], which will induce two types of LieBN: one w.r.t. left-invariance and another w.r.t. right-invariance. On the correlation manifold, we manifest our LieBN under four recently developed correlation geometries [195, 191]. To facilitate usage, we provide a LieBN toolbox compatible with PyTorch, which can be used as a drop-in module. Fig. 3.1 illustrates our LieBN on different geometries, while Fig. 3.2 illustrates a minimal demo. Extensive experiments on SPD, rotation, and correlation manifolds involving radar recognition, human action recognition, and electroencephalography (EEG) classification demonstrate the effectiveness of our methods.

We emphasize that our work is fundamentally distinct from Brooks et al. [31], Kobler et al. [123], Lou et al. [142] in theory and more general than Chakraborty [34]. Previous RBN methods are either designed for specific geometries [31, 123, 34] or fail to control both the mean and variance [142]. In contrast, our LieBN ensures the normalization of both the mean and variance across general Lie groups. In summary, our main contributions are:

- A general LieBN framework with controllable first- and second-order moments;
- A novel right-invariant metric on the SPD manifold, which is the first non-trivial right-invariant SPD metric;
- Concrete instantiations of our LieBN framework on different geometries: four on SPD manifolds, one on rotation matrices, and four on correlation manifolds;

¹Although some metrics are bi-invariant, the associated group structures are commutative [9, 137]. Therefore, their bi-invariance is reduced to left-invariance.

```

from LieBN import LieBNSPD, LieBNRot, LieBNCor
from LieBN.Geometry.SPD import SPDMatrices
from LieBN.Geometry.Rotations import RotMatrices
from LieBN.Geometry.Correlation import Correlation

# ==== SPD matrices ====
P_spd = SPDMatrices(n=5).random(4, 2, 5, 5)
# Implemented metrics: LEM, ALEM, LCM, AIM, CRIM
liebn_spd = LieBNSPD([2, 5, 5], metric="LEM", batchdim=[0])
output_spd = liebn_spd(P_spd)

# ==== SO(3) matrices ====
P_so3 = RotMatrices().random(4, 2, 3, 3, 3)
# LieBN-Left if is_left else -Right
liebn_so3 = LieBNRot([3, 3, 3], batchdim=[0, 1], is_left=False)
output_so3 = liebn_so3(P_so3)

# ==== Correlation matrices ====
P_cor = Correlation(n=5).random(4, 2, 5, 5)
# Implemented metrics: ECM, LECM, OLM, LSM
liebn_cor = LieBNCor([2, 5, 5], metric="ECM", batchdim=[0])
output_cor = liebn_cor(P_cor)

```

Figure 3.2: Minimal examples of applying LieBN.

- Validation of the effectiveness of our LieBN framework by extensive experiments on different geometries.²

Outline. Sec. 3.2.2 recalls the invariant metrics and Lie structures used by LieBN. Sec. 3.2.3 revisits Euclidean BN and RBN. Sec. 3.2.4 develops LieBN on Lie groups under left- and right-invariant metrics and establishes its statistical control. Sec. 3.2.5 instantiates LieBN on SPD, rotation, and full-rank correlation manifolds. Sec. 3.2.6 reports experiments that validate LieBN across these geometries. Proofs are deferred to Sec. B.2.

3.2.2 Preliminaries

An invariant metric can be understood as the Lie-group analogue of the Euclidean inner product being unaffected by translations. In Euclidean space, adding the same vector on the left or on the right preserves inner products, while on a Lie group the corresponding requirement is that left or right group translations preserve the Riemannian metric.

²The code is available at <https://github.com/GitZH-Chen/LieBN.git>.

Operator	(α, β) -AIM	(α, β) -LEM	LCM
$Q \oplus P$	$KPK^\top$	$\exp(\log(P) + \log(Q))$	$\text{Chol}^{-1}(\lfloor L + K \rfloor + \mathbb{KL})$
$\ominus P$	$\text{Chol}^{-1}(L^{-1})$	$\exp(-\log(P))$	$\text{Chol}^{-1}(-\lfloor L \rfloor + \mathbb{L}^{-1})$
Identity	I_n	I_n	I_n
WFM	Karcher Flow	$\exp(\sum_i w_i \log(P_i))$	$\psi_{\text{LC}}^{-1}(\sum_i w_i \psi_{\text{LC}}(P_i))$
Invariance	Left-invariance	Bi-invariance	Bi-invariance

Table 3.1: Review of SPD Lie groups and invariant metrics.

Group	$Q \oplus P$	$\ominus P$	Identity	WFM	Invariance
$\text{SO}(n)$	QP	$P^{-1} = P^\top$	I_n	Karcher Flow	Bi-invariance

Table 3.2: Review of the rotation Lie group and invariant metric.

Definition 60 (Invariance [69]). A Riemannian metric g^L over a Lie group $\{\mathcal{M}, \oplus\}$ is left-invariant if, for any $x, y \in \mathcal{M}$ and $V_1, V_2 \in T_y \mathcal{M}$, it satisfies $g_y^L(V_1, V_2) = g_{L_x(y)}^L(L_{x*,y}(V_1), L_{x*,y}(V_2))$, with $L_x(y) = x \oplus y$ as the left translation by x , and $L_{x*,y}$ as the differential map of L_x at y . Similarly, a right-invariant metric g^R satisfies $g_y^R(V_1, V_2) = g_{R_x(y)}^R(R_{x*,y}(V_1), R_{x*,y}(V_2))$, with $R_x(y) = y \oplus x$ as the right translation by x , and $R_{x*,y}$ as the differential map of R_x at y .

Many popular matrix manifolds used in machine learning form Lie groups, including the SPD manifold, the full-rank correlation manifold, and the rotation group. Although the corresponding Riemannian geometries have been reviewed in Secs. 2.9.1, 2.9.2 and 2.9.4, Tabs. 3.1 to 3.3 provide a focused recap of their Lie structures. The notation follows Tabs. 2.5, 2.7 to 2.9 and 2.12.

3.2.3 Revisiting Normalization

3.2.3.1 Revisiting Euclidean Normalization

In Euclidean DNNs, normalization is a significant technique for accelerating network training by mitigating the issue of internal covariate shift [109]. While various normalization methods have been introduced [109, 12, 198, 219], they all share a common purpose: the normalization of the first and second moments. We focus on BN, the prototype of other normalization variants.

Given a batch of activations $\{x_i\}_{i=1}^N$, the core operations in the standard Euclidean

3.2. Lie Group Batch Normalization

Operator	ECM	LECM	OLM	LSM
$C \oplus C'$		$\phi^{-1}(\phi(C) + \phi(C'))$		
$\ominus C$		$\phi^{-1}(-\phi(C))$		
Identity		$\phi^{-1}(\mathbf{0}_{n \times n})$		
WFM		$\phi^{-1}\left(\sum_{i=1}^N w_i \phi(C_i)\right)$		
Invariance		Bi-invariance		

Table 3.3: Review of full-rank correlation Lie groups and invariant metrics. Here ϕ denotes Θ , $\log \circ \Theta$, Log° , and Log^* for ECM, LECM, OLM, and LSM, respectively.

Methods	Involved Statistics	Controllable Mean	Controllable Variance	Geometries
SPDBN [31, Alg. 1]	Mean	✓	N/A	SPD manifolds under AIM
SPDBN [124, Alg. 1]	Mean+Variance	✓	✓	SPD manifolds under AIM
SPDDSMBN [123]	Mean+Variance	✓	✓	SPD manifolds under AIM
ManifoldNorm [34, Algs. 1-2]	Mean+Variance	✗	✗	Riemannian homogeneous spaces
ManifoldNorm [34, Algs. 3-4]	Mean+Variance	✓	✓	A specific Lie group structure and distance
RBN [142, Alg. 2]	Mean+Variance	✗	✗	Geodesically complete manifolds
LieBN (Ours)	Mean+Variance	✓	✓	Lie groups

Table 3.4: Summary of some representative RBN methods.

BN can be expressed as:

$$\forall i \leq N, x_i \leftarrow \gamma \frac{x_i - \mu_b}{\sqrt{v_b^2 + \epsilon}} + \beta \quad (3.1)$$

where μ_b is the batch mean, v_b^2 is the batch variance, γ is the scaling parameter, β is the biasing parameter, and ϵ is a small scalar for stability.

3.2.3.2 Revisiting RBN

Although endeavors have been made to develop Riemannian normalization approaches tailored for manifolds, none of the existing methods effectively handle the first and second moments in a principled manner.

Brooks et al. [31] introduced RBN over SPD manifolds under AIM. The core operations are defined as follows:

$$\text{Centering from mean } M \in \mathcal{S}_{++}^n : \bar{P}_i \leftarrow M^{-\frac{1}{2}} P_i M^{-\frac{1}{2}}, \quad (3.2)$$

$$\text{Biasing towards parameter } B \in \mathcal{S}_{++}^n : \hat{P}_i \leftarrow B^{\frac{1}{2}} \bar{P}_i B^{\frac{1}{2}}, \quad (3.3)$$

where $\{P_i\}_{i=1}^N$ are SPD matrices, and M is their Fréchet mean under AIM. Let

$$\Gamma_{P \rightarrow Q}(S) = \text{Exp}_Q [\text{PT}_{P \rightarrow Q}(\text{Log}_P(S))], \quad (3.4)$$

where $P, Q, S \in \mathcal{S}_{++}^n$. Under AIM, Eqs. (3.2) and (3.3) can be more generally expressed as

$$\Gamma_{I \rightarrow B}[\Gamma_{M \rightarrow I}(P_i)]. \quad (3.5)$$

However, Eqs. (3.2) and (3.3) only consider the Riemannian mean³ and do not consider the Riemannian variance. To remedy this limitation, Kobler et al. [123] further extended the RBN to involve the second-order statistics. The key operation is formulated as

$$\forall i \leq N, \bar{P}_i \leftarrow \Gamma_{I \rightarrow B}[(\Gamma_{M \rightarrow I}(P_i))^{\frac{s}{v}}], \quad (3.6)$$

where v^2 is the Fréchet variance, and $s \in \mathbb{R}$ is a scaling factor. However, this method is still limited to SPD manifolds under AIM. In parallel, Chakraborty [34, Algs. 1–2] proposed a general framework for Riemannian homogeneous spaces based on Eq. (3.5), which involves both first and second moments. However, Eq. (3.5) does not generally guarantee control over the Riemannian mean, resulting in agnostic Riemannian statistics [34, Sec. 3.1]. To mitigate this limitation, Chakraborty [34, Algs. 3–4] further proposed normalization over matrix Lie groups. However, the discussion is limited to a certain distance, limiting the applicability of their method. On the other hand, Lou et al. [142, Alg. 2] proposed an RBN based on a variant of Eq. (3.5). Similarly, their approach suffers from the same problem of agnostic Riemannian statistics on general manifolds.

In summary, prevailing Riemannian normalization approaches lack a principled guarantee for controlling the first- and second-order statistics. In contrast, our method can normalize first- and second-order statistics over general Lie groups. We summarize the above RBN methods in Tab. 3.4.

3.2.4 LieBN

Since every Lie group naturally admits invariant metrics, we propose BN over Lie groups based on invariant metrics, referred to as LieBN. We first introduce the core operations under left-invariant metrics and then extend them to right-invariant metrics. Finally, we present the theoretical LieBN framework. In the following, we denote the neutral element in the Lie group $\mathcal{M}$ as E ⁴.

³Although not discussed in Brooks et al. [31], the congruent actions in Eqs. (3.2) and (3.3) can transfer the batch mean to a desired value under AIM.

⁴The neutral element E is not necessarily the identity matrix.

3.2.4.1 Ingredients under Left-invariant Metrics

In this subsection, we always assume that the Lie group $\mathcal{M}$ admits a left-invariant metric g^L . Recalling the standard Euclidean BN [109] in Eq. (3.1), two key points are noteworthy: (a) the Euclidean BN implicitly assumes a Gaussian distribution and can effectively normalize the latent Gaussian distribution; (b) the centering and biasing operations control the mean, while the scaling controls the variance. Therefore, extending BN to Lie groups requires Lie-group counterparts of the Gaussian distribution, centering, biasing, and scaling.

There are several notions of Gaussian distribution over manifolds [170, 218, 35, 14]. We adopt the intrinsic definition from Chakraborty and Vemuri [35], which characterizes a Gaussian distribution on the Lie group $\mathcal{M}$ with a mean parameter $M \in \mathcal{M}$ and variance σ^2 . This distribution is denoted as $\mathcal{N}(M, \sigma^2)$, and its probability density function (PDF) is

$$p(X | M, \sigma^2) = k(\sigma) \exp\left(-\frac{d(X, M)^2}{2\sigma^2}\right), \quad (3.7)$$

where $k(\sigma)$ is the normalizing constant and $d(\cdot, \cdot)$ is the geodesic distance. When $\mathcal{M}$ is $\mathbb{R}$ with the standard Euclidean metric, Eq. (3.7) reduces to the Euclidean Gaussian.

On Lie groups, the natural counterparts of addition and subtraction in Eq. (3.1) are group operations. Therefore, centering and biasing on Lie groups can be defined by the left translation. Additionally, we define scaling via the tangent space. Specifically, for a batch of activations $\{P_i\}_{i=1}^N \subset \mathcal{M}$, we define the key operations of LieBN as follows:

$$\text{Centering from mean } M \in \mathcal{M} : \bar{P}_i \leftarrow L_{\ominus M}(P_i), \quad (3.8)$$

$$\text{Scaling: } \hat{P}_i \leftarrow \text{Exp}_E \left[\frac{s}{\sqrt{v^2 + \epsilon}} \text{Log}_E(\bar{P}_i) \right], \quad (3.9)$$

$$\text{Biasing towards parameter } B \in \mathcal{M} : \tilde{P}_i \leftarrow L_B(\hat{P}_i), \quad (3.10)$$

where M is the Fréchet mean, v^2 is the Fréchet variance, $\ominus M \in \mathcal{M}$ is the group inverse of M , $L_{\ominus M}$ and L_B are left translations ($L_B(P_i) = B \oplus P_i$), and $s \in \mathbb{R} \setminus \{0\}$ is a scaling parameter. The following two propositions demonstrate the above operations in normalizing mean and variance: one related to population statistics and the other related to sample statistics.

Proposition 61 (Population). [↓] *Given a random point X over $\{\mathcal{M}, \oplus, g^L\}$, and the Gaussian distribution $\mathcal{N}(M, v^2)$ defined in Eq. (3.7), we have the following for*

the population statistics:

- (1) (MLE of M) Given $\{P_i\}_{i=1}^N \subset \mathcal{M}$ i.i.d. sampled from $\mathcal{N}(M, v^2)$, the maximum likelihood estimator (MLE) of M is the sample Fréchet mean.
- (2) (Gaussian homogeneity) Given $X \sim \mathcal{N}(M, v^2)$ and $B \in \mathcal{M}$, we have

$$L_B(X) \sim \mathcal{N}(L_B(M), v^2). \quad (3.11)$$

Proposition 62 (Sample). [↓] Given N samples $\{P_i\}_{i=1}^N$ over the Lie group $\{\mathcal{M}, \oplus, g^L\}$, define

$$\phi_s(P_i) = \text{Exp}_E[s \text{Log}_E(P_i)]. \quad (3.12)$$

We then have the following for the sample statistics.

- Sample mean homogeneity:

$$\text{FM}\{L_B(P_i)\} = L_B(\text{FM}\{P_i\}), \forall B \in \mathcal{M}. \quad (3.13)$$

- Controllable dispersion from E :

$$\sum_{i=1}^N w_i d^2(\phi_s(P_i), E) = s^2 \sum_{i=1}^N w_i d^2(P_i, E), \quad (3.14)$$

where $\{w_i\}_{i=1}^N$ are weights satisfying a convexity constraint, i.e., $\forall i, w_i > 0$ and $\sum_i w_i = 1$.

Thm. 61 and Eq. (3.13) imply that our centering and biasing in Eqs. (3.8) and (3.10) can transfer the sample and population mean. As the post-centering mean is E , Eq. (3.14) implies that Eq. (3.9) can control the sample variance. More interestingly, the latent Gaussian distribution can be transferred under some geometries, such as SPD manifolds under LEM and LCM.

Remark 63. The MLE of the mean of the Gaussian distribution has been examined in several previous works [176, 35, 34]. However, these studies primarily focus on particular manifolds or specific metrics. In contrast, our contribution lies in presenting a general result for Lie groups.

Remark 64. While Eq. (3.7) appeared in Kobler et al. [124], the authors only focus on SPD manifolds under AIM. The transformation of the population under their proposed RBN remains unexplored as well. Besides, while Chakraborty [34] analyzed the population properties for their RBN over matrix Lie groups, their results were confined within a specific distance. In contrast, our work provides a more extensive examination, encompassing both population and sample properties of our LieBN in a general manner.

3.2.4.2 Ingredients under Right-invariant Metrics

The key insight beneath Eqs. (3.8) and (3.10) and Thms. 61 and 62 is that left translation is an isometry under left-invariant metrics. Similarly, right translation is an isometry under right-invariant metrics. Therefore, it can be used for centering and biasing under right-invariant metrics. Following the previous notations, we define the centering and biasing under a right-invariant metric g^R as

$$\text{centering to } E: \bar{P}_i \leftarrow R_{\ominus M}(P_i), \quad (3.15)$$

$$\text{biasing towards } B: \tilde{P}_i \leftarrow R_B(\hat{P}_i). \quad (3.16)$$

Similar to the case under left-invariant metrics, Thms. 61 and 62 can be easily extended to right-invariant metrics. Notably, the proofs for the MLE of M in Thm. 61 and controllable dispersion in Thm. 62 can be directly applied to the right-invariant metric. Therefore, we only show the homogeneity in the following proposition.

Proposition 65. [↓] *Given a random point $X \sim \mathcal{N}(M, v^2)$ over $\{\mathcal{M}, \oplus, g^R\}$, $B \in \mathcal{M}$, and N samples $\{P_i\}_{i=1}^N$ over $\mathcal{M}$, we have:*

- (1) *Gaussian homogeneity:* $R_B(X) \sim \mathcal{N}(R_B(M), v^2)$;
- (2) *Sample homogeneity:* $\text{FM}\{R_B(P_i)\} = R_B(\text{FM}\{P_i\})$.

3.2.4.3 LieBN under Invariant Metrics

With the above ingredients, Alg. 1 presents our theoretical LieBN framework. Similar to Ioffe and Szegedy [109], we use the moving average to update the running statistics. For a bi-invariant metric, LieBN can be implemented using either left or right translation. If the Lie group is commutative, LieBN under left and right translations are equivalent. Tab. 3.5 summarizes the LieBN types under different conditions.

Commutativity	Non-commutative			Commutative
Invariance	Left	Right	Bi	Left = Right = Bi
LieBN Types	Left	Right	Left & Right	Left = Right

Table 3.5: Summary of LieBN types.

The centering and biasing in Euclidean BN correspond to the group action of $\mathbb{R}$. From a geometric perspective, the standard Euclidean metric is invariant under this group operation. Consequently, it is not surprising that our LieBN algorithm naturally generalizes the standard Euclidean BN.

Proposition 66. $\Downarrow$ *The LieBN algorithm presented in Alg. 1 is equivalent to the standard Euclidean BN when $\mathcal{M} = \mathbb{R}^n$, both during the training and testing phases.*

3.2.5 Manifestations

This section instantiates our LieBN in Alg. 1 on nine different Lie groups, including four on the SPD manifold, one on rotation matrices, and four on the correlation manifold.

3.2.5.1 LieBN on SPD Manifolds

We first extend the current Lie groups on SPD manifolds by the matrix power deformation, resulting in three families of parameterized Lie groups. Then, we propose a novel right-invariant metric on the SPD manifold, the first non-trivial right-invariant metric on this manifold. Finally, we construct LieBN layers based on these Lie structures.

Deformed Lie structures on SPD manifolds. As shown in Tab. 3.1, there are three Lie groups on SPD manifolds, each with a left-invariant metric. These metrics include (α, β) -AIM, (α, β) -LEM, and LCM. For clarity, we denote the group operations w.r.t. (α, β) -AIM, (α, β) -LEM and LCM as $\oplus^{\text{LieAI}}$, $\oplus^{\text{LE}}$ and $\oplus^{\text{LC}}$, respectively.

Recently, Thanwerdas and Pennec [193] further extended (α, β) -AIM into a three-parameter family of metrics via the pullback of the matrix power function $P_\theta(\cdot)$, scaled by $\frac{1}{\theta^2}$ and denoted by (θ, α, β) -AIM. The matrix power serves as a deformation, wherein (θ, α, β) -AIM encompasses (α, β) -AIM with $\theta = 1$, and becomes (α, β) -LEM as θ approaches 0 [192]. Inspired by the deforming utility of the power function, we define the power-deformed metrics of (α, β) -LEM and LCM as the pullback metrics by P_θ and scaled by $\frac{1}{\theta^2}$. We denote these two metrics as (θ, α, β) -LEM and θ -LCM, respectively. We have the following results with respect to the deformation.

Algorithm 1: Lie Group Batch Normalization (LieBN)

Input : A batch of activations $\{P_i\}_{i=1}^N$ over Lie groups $\{\mathcal{M}, \oplus, g\}$, a small positive constant ϵ , and momentum $\eta \in [0, 1]$, running mean $M_r = E$, running variance $v_r^2 = 1$, biasing parameter $B \in \mathcal{M}$, and scaling parameter $s \in \mathbb{R} \setminus \{0\}$.

Output : Normalized activations $\{\tilde{P}_i\}_{i=1}^N$.

if *training* **then**

- Compute batch mean M_b and variance v_b^2
- Update running statistics: $M_r \leftarrow \text{WFM}(\{1 - \eta, \eta\}, \{M_r, M_b\})$
- $v_r^2 \leftarrow (1 - \eta)v_r^2 + \eta v_b^2$
- Use the batch statistics, $M \leftarrow M_b, v^2 \leftarrow v_b^2$

else

- Use the running statistics, $M \leftarrow M_r, v^2 \leftarrow v_r^2$

for $i \leftarrow 1$ **to** N **do**

- Centering to the neutral element E :
 - if** g is left-invariant **then**
 - $\bar{P}_i \leftarrow L_{\ominus M}(P_i)$
 - else**
 - $\bar{P}_i \leftarrow R_{\ominus M}(P_i)$
- Scaling the variance:
 - $\hat{P}_i \leftarrow \text{Exp}_E \left[\frac{s}{\sqrt{v^2 + \epsilon}} \text{Log}_E(\bar{P}_i) \right]$
- Biasing towards parameter B :
 - if** g is left-invariant **then**
 - $\tilde{P}_i \leftarrow L_B(\hat{P}_i)$
 - else**
 - $\tilde{P}_i \leftarrow R_B(\hat{P}_i)$

Proposition 67 (Deformation). $\llbracket \downarrow \rrbracket$ (θ, α, β) -LEM is equal to (α, β) -LEM. θ -LCM interpolates between $\tilde{g}$ -LEM (as $\theta \rightarrow 0$) and LCM ($\theta = 1$). Here, given any $P \in \mathcal{S}_{++}^n$ and tangent vectors $V, W \in T_P \mathcal{S}_{++}^n$, $\tilde{g}$ -LEM is defined as

$$\langle V, W \rangle_P = \tilde{g}(\log_{*,P}(V), \log_{*,P}(W)), \quad (3.17)$$

where $\tilde{g}(V_1, V_2) = \frac{1}{2} \langle V_1, V_2 \rangle - \frac{1}{4} \langle \mathbb{D}(V_1), \mathbb{D}(V_2) \rangle$, $\mathbb{D}(V_i)$ is a diagonal matrix consisting of the diagonal elements of V_i , and $\log_{*,P}$ is the differential map at P .

As (θ, α, β) -LEM is equal to (α, β) -LEM, we focus on (α, β) -LEM, (θ, α, β) -AIM, and θ -LCM in the following. As a diffeomorphism, P_θ can also pull back the group operations $\oplus^{\text{LieAI}}$ and $\oplus^{\text{LC}}$, denoted by $\oplus^{\theta\text{-AI}}$ and $\oplus^{\theta\text{-LC}}$, respectively. We have the

following proposition on the invariance.

Proposition 68 (Invariance). $\llbracket \downarrow \rrbracket$ (θ, α, β) -AIM is left-invariant w.r.t. $\oplus^{\theta-AI}$, while θ -LCM is bi-invariant w.r.t. $\oplus^{\theta-LC}$.

SPD right-invariant metrics. AIM is left-invariant w.r.t. $\oplus^{\text{LieAI}}$. We can also define a right-invariant metric w.r.t. $\oplus^{\text{LieAI}}$ by definition [69, Ch. 1.2]:

$$g_P^{\text{CRI}}(V, W) = \langle (R_{\ominus^{\text{LieAI}}P})_{*,P}(V), (R_{\ominus^{\text{LieAI}}P})_{*,P}(W) \rangle_I \quad (3.18)$$

where $R_{(\cdot)}$ denotes Lie group right translation, $\ominus^{\text{LieAI}}P$ is the inverse of P under $\oplus^{\text{LieAI}}$, and $\langle \cdot, \cdot \rangle_I$ denotes an arbitrary inner product on $T_I \mathcal{S}_{++}^n$. We set $\langle \cdot, \cdot \rangle_I$ to be the same as the AIM at I , i.e., $\langle \cdot, \cdot \rangle^{(\alpha, \beta)}$. We call this metric CRIM, as the group operation is defined by the matrix product of Cholesky factors [195, Sec. 3.2].

Theorem 69. $\llbracket \downarrow \rrbracket$ Given any SPD matrices P, Q and tangent vector $V \in T_P \mathcal{S}_{++}^n$, the Riemannian operators on $\{\mathcal{S}_{++}^n, g^{\text{CRI}}\}$ are

$$g_P^{\text{CRI}}(V, V) = \left(\left\| \left(L(L^{-1}VL^{-\top})_{\frac{1}{2}}L^{-1} \right)_{\text{sym}+} \right\|^{(\alpha, \beta)} \right)^2 \quad (3.19)$$

$$d(P, Q) = \left\| \log \left(\tilde{Q}^{-\frac{1}{2}} \tilde{P} \tilde{Q}^{-\frac{1}{2}} \right) \right\|^{(\alpha, \beta)}, \quad (3.20)$$

$$\text{Exp}_P(V) = \ominus^{\text{LieAI}} \left(\text{Exp}_{\tilde{P}}^{\text{AI}}(-\tilde{V}) \right), \quad (3.21)$$

$$\text{Log}_P(Q) = - \left(LL^{\top} \left(L \tilde{V} L^{\top} \right)_{\frac{1}{2}}^{\top} \right)_{\text{sym}+}, \quad (3.22)$$

where L is the Cholesky factor of $P = LL^{\top}$, $\ominus^{\text{LieAI}}(\cdot)$ is the group inverse, $\tilde{P}$ and $\tilde{Q}$ are the group inverses of P and Q , $\tilde{V} = \left((L^{-1}VL^{-\top})_{\frac{1}{2}} L^{-1} L^{-\top} \right)_{\text{sym}+}$, and $\tilde{V} = \text{Log}_{\tilde{P}}^{\text{AI}}(\tilde{Q})$. Here, $(X)_{\text{sym}+} = X + X^{\top}$, $\forall X \in \mathbb{R}^{n \times n}$ denotes unnormalized symmetrization, and $(X)_{\frac{1}{2}} = \lfloor X \rfloor + \frac{1}{2}\mathbb{X}$.

Corollary 70. $\llbracket \downarrow \rrbracket$ CRIM is geodesically complete, and the associated geodesic connecting SPD matrices P and Q is

$$\begin{aligned} \gamma_{(P,Q)}(t) &= \ominus^{\text{LieAI}} \left\{ \gamma^{\text{AI}}(t; \tilde{P}, \tilde{Q}) \right\} \\ &= \ominus^{\text{LieAI}} \left\{ \tilde{P}^{\frac{1}{2}} \left(\tilde{P}^{-\frac{1}{2}} \tilde{Q} \tilde{P}^{-\frac{1}{2}} \right)^t \tilde{P}^{\frac{1}{2}} \right\}, \end{aligned} \quad (3.23)$$

3.2. Lie Group Batch Normalization

Metric	(θ, α, β) -AIM	(α, β) -LEM	θ -LCM	θ -CRIM
Invariance	Left-invariance	Bi-invariance		Right-invariance
LieBN Type	LieBN-Left	LieBN-Left = LieBN-Right		LieBN-Right
Pullback Map	P_θ	$\log$	$P_\theta \circ \psi_{LC}$	P_θ
Codomain	$\{\mathcal{S}_{++}^n, \oplus^{\text{LieAI}}, \frac{1}{\theta^2} g^{(\alpha, \beta)\text{-AI}}\}$	$\{\mathcal{S}^n, \langle \cdot, \cdot \rangle^{(\alpha, \beta)}\}$	$\{\text{LT}^n, \frac{1}{\theta^2} \langle \cdot, \cdot \rangle\}$	$\{\mathcal{S}_{++}^n, \oplus^{\text{LieAI}}, \frac{1}{\theta^2} g^{\text{CRIM}}\}$
Riemannian and Lie group operators in the codomain	$L_Q(P)$ or $R_Q(P)$	$KPK^\top$	$P + Q$	$P + Q$
	$L_{\ominus Q}(P)$ or $R_{\ominus Q}(P)$	$K^{-1}PK^{-\top}$	$P - Q$	$P - Q$
	$\text{Exp}_E[s \text{Log}_E(P)]$	P^s	sP	sP
	FM	Karcher Flow	Arithmetic average	Arithmetic average
	WFM($\{1 - \eta, \eta\}, \{P_1, P_2\}$)	$P_1^{\frac{1}{2}} \left(P_1^{-\frac{1}{2}} P_2 P_1^{-\frac{1}{2}} \right)^\eta P_1^{\frac{1}{2}}$	Arithmetic weighted average	Arithmetic weighted average

Table 3.6: Key operators in calculating LieBN on SPD manifolds.

Invariance	LieBN Type	$\ominus R$	$L_R(S)$	$R_R(S)$	$\text{Exp}_I[s \text{Log}_I(R)]$	FM	WFM($\{1 - \eta, \eta\}, \{R, S\}$)
Bi-invariance	LieBN-Left & LieBN-Right	R^{-1}	RS	SR	$\exp(s \log(R))$	[146, Alg. 1]	$R \exp(\eta \log(R^\top S))$

Table 3.7: Key operators in calculating LieBN on the rotation matrices.

where $\tilde{P} = \ominus^{\text{LieAI}} P$ and $\tilde{Q} = \ominus^{\text{LieAI}} Q$ are group inverses, with γ^{AI} as the geodesic under AIM.

Similar to the discussion in Sec. 3.2.5.1, we define θ -CRIM as the deformed metric of CRIM by the pullback of matrix power function $P_\theta(\cdot)$ and scaled by $\frac{1}{\theta^2}$. As the pullback of CRIM, θ -CRIM is right-invariant w.r.t. $\oplus^{\theta\text{-AI}}$ by definition.

Proposition 71. θ -CRIM is right-invariant w.r.t. $\oplus^{\theta\text{-AI}}$.

Manifestations on SPD manifolds. So far, there are four families of invariant metrics on the SPD Lie groups: (1) left-invariant (θ, α, β) -AIM w.r.t. $\oplus^{\text{LieAI}}$; (2) bi-invariant (α, β) -LEM w.r.t. $\oplus^{\text{LE}}$ and θ -LCM w.r.t. $\oplus^{\text{LC}}$; (3) right-invariant θ -CRIM w.r.t. $\oplus^{\text{LieAI}}$. Since all the above metrics are pullback metrics, the LieBN based on these metrics can be simplified and calculated in the codomain. We first show a general result on LieBN under the pullback metric. We denote Alg. 1 on the Lie group $\mathcal{M}$ as

$$\text{LieBN}(P_i; B, s, \epsilon, \eta), \quad P_i \in \{P_j\}_{j=1}^N \subset \mathcal{M}. \quad (3.24)$$

Then we can obtain the following theorem.

Theorem 72. [↓] Given a Lie group $\mathcal{M}_1$, a Lie group $\mathcal{M}_2$ with an invariant metric g^2 , and a map $f : \mathcal{M}_1 \rightarrow \mathcal{M}_2$ that is both a diffeomorphism and a Lie-group isomorphism, the map f induces an invariant metric g^1 on $\mathcal{M}_1$, denoted as $g^1 = f^* g^2$. For a batch of activations $\{P_i\}_{i=1}^N$ in $\mathcal{M}_1$, $\text{LieBN}^1(P_i; B, s, \epsilon, \eta)$ in $\mathcal{M}_1$ can

be calculated in $\mathcal{M}_2$ by the following process:

$$\text{Mapping data into } \mathcal{M}_2 : \bar{P}_i = f(P_i), \bar{B} = f(B), \quad (3.25)$$

$$\text{Performing LieBN in } \mathcal{M}_2 : \hat{P}_i = \text{LieBN}^2(\bar{P}_i; \bar{B}, s, \epsilon, \eta), \quad (3.26)$$

$$\text{Mapping the resulting data back to } \mathcal{M}_1 : \tilde{P}_i = f^{-1}(\hat{P}_i), \quad (3.27)$$

where LieBN^2 is the LieBN on $\mathcal{M}_2$.

Given a metric g on $\mathcal{S}_{++}^n$, the power-deformed metric $\tilde{g} = \frac{1}{\theta^2} P_\theta^* g$ is equal to $P_\theta^*(\frac{1}{\theta^2} g)$. Thm. 72 indicates that the LieBN under $\tilde{g}$ can be calculated by the LieBN under $\frac{1}{\theta^2} g$. Besides, as the Christoffel symbols remain the same under constant scaling, the LieBNs under $\frac{1}{\theta^2} g$ and g only differ in the variance. We denote $g^{(\alpha, \beta)\text{-AI}}$ and $g^{(\theta, \alpha, \beta)\text{-AI}}$ as the metric tensors of (α, β) -AIM and (θ, α, β) -AIM, respectively. Based on the above discussions, the computations of the LieBN under $g^{(\theta, \alpha, \beta)\text{-AI}}$ are reduced to the LieBN under $\frac{1}{\theta^2} g^{(\alpha, \beta)\text{-AI}}$. Similarly, denoting g^{CRIM} and $g^{\theta\text{-CRIM}}$ as the metric tensors of CRIM and θ -CRIM, then the LieBN under θ -CRIM can be calculated by the one under $\frac{1}{\theta^2} g^{\text{CRIM}}$. Furthermore, as shown in Sec. 4.2.3, (α, β) -LEM is a pullback metric from the Euclidean space $\mathcal{S}^n$ of symmetric matrices, while θ -LCM is a pullback metric from the Euclidean space LT^n of lower triangular matrices. As shown in Thm. 66, the LieBN in the Euclidean space $\mathcal{S}^n$ or LT^n is simplified to the standard Euclidean BN. Therefore, the LieBNs under (α, β) -LEM and θ -LCM can be calculated by the Euclidean BN over $\mathcal{S}^n$ and LT^n , respectively.

We denote the LieBN under left and right translations as LieBN-Left and LieBN-Right, respectively. Then, the LieBNs under (θ, α, β) -AIM and θ -CRIM correspond to LieBN-Left and LieBN-Right, respectively. As $\oplus^{\text{LC}}$ and $\oplus^{\text{LE}}$ are commutative, the LieBN-Left and LieBN-Right under (α, β) -LEM and θ -LCM are equivalent. We denote P, Q, P_1 and P_2 as points in the codomain, i.e., $\mathcal{S}_{++}^n$ with scaled CRIM for θ -CRIM, $\mathcal{S}_{++}^n$ with scaled (α, β) -AIM for (θ, α, β) -AIM, $\mathcal{S}^n$ for (α, β) -LEM, and LT^n for θ -LCM, respectively. For CRIM, we denote $\tilde{P}_i = \ominus^{\text{LieAI}} P_i$ for $i = 1, 2$. We summarize all the necessary ingredients in Tab. 3.6 for calculating SPD LieBN. Note that for (θ, α, β) -AIM, our scaling operation defined in Eq. (3.9) encompasses the scaling operation in Kobler et al. [124, Eq. (9)] as a special case, when $(\theta, \alpha, \beta) = (1, 1, 0)$.

3.2.5.2 LieBN on Rotation Matrices

As the Riemannian metric on the rotation matrices is bi-invariant, there are two instantiations of LieBN on this manifold, i.e., LieBN-Left based on the left translation

3.2. Lie Group Batch Normalization

Metric	ECM	LECM	OLM	LSM
Invariance	Bi-invariance			
LieBN Type	LieBN-Left = LieBN-Right			
Pullback Map	Θ	$\log \circ \Theta$	Log°	$\text{Log}^\star$
Codomain	$\{\text{LT}^1(n), \langle \cdot, \cdot \rangle\}$	$\{\text{LT}^0(n), \langle \cdot, \cdot \rangle\}$	$\{\text{Hol}(n), \langle \cdot, \cdot \rangle^{(\alpha, \beta, \gamma)}\}$	$\{\text{Row}_0(n), \langle \cdot, \cdot \rangle^{(\alpha, \delta, \zeta)}\}$

Table 3.8: Summary of LieBN on the correlation. $\langle \cdot, \cdot \rangle^{(\alpha, \beta, \gamma)}$ and $\langle \cdot, \cdot \rangle^{(\alpha, \delta, \zeta)}$ are permutation-invariant inner products [191].

and LieBN-Right based on the right translation. In particular, the scaling can be further simplified: $\text{Exp}_I(s \text{Log}_I(R)) = \exp(s \log(R))$. For the specific $\text{SO}(3)$, the matrix $\exp$ and $\log$ can be efficiently calculated without matrix decomposition [95, Sec. 3.2]. Tab. 3.7 presents the expressions of the required operators in Alg. 1.

3.2.5.3 LieBN on Full-Rank Correlation Matrices

As summarized in Tab. 3.3, all four correlation metrics are bi-invariant, and their associated Lie groups are commutative. Consequently, LieBN-Left is identical to LieBN-Right. Moreover, all four correlation metrics are pullback metrics from simpler Euclidean spaces. Therefore, LieBN over the correlation can be implemented according to Thm. 72: (1) map the correlation into the prototype Euclidean space, (2) apply Euclidean BN, and (3) map back to the correlation.

Optimization. Finally, we discuss the optimization of the correlation-valued biasing parameter $B \in \text{Cor}^+(n)$. As reviewed in Sec. 2.9.2, the correlation matrix can be identified by the product of hyperbolic spaces via the Cholesky decomposition. Given $C \in \text{Cor}^+(n)$, the k -th row of the Cholesky factor $L = \text{Chol}(C)$ is $(L_{k1}, \dots, L_{k,k-1}, L_{kk}, 0, \dots, 0)$ with $L_{kk} > 0$, which belongs to the hyperbolic space of an open hemisphere

$$\text{HS}^{k-1} = \{x \in \mathbb{R}^k \mid \|x\| = 1, x_k > 0\}. \quad (3.28)$$

Besides, the open hemisphere HS^n is isometric to the Poincaré ball

$$\mathbb{P}^n = \{x \in \mathbb{R}^n \mid \|x\| < 1\} \quad (3.29)$$

by

$$\pi_{\text{HS}^n \rightarrow \mathbb{P}^n}((x^\top, x_{n+1})^\top) = \frac{x}{1 + x_{n+1}}. \quad (3.30)$$

Therefore, each correlation can be parameterized with $n - 1$ Poincaré vectors. Each

Poincaré vector can be optimized directly on its manifold using the Riemannian optimization strategy reviewed in Sec. 2.7. The above process can be expressed as

$$C \mapsto \begin{pmatrix} 1 & 0 & \cdots & 0 \\ L_{21} & L_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n1} & L_{n2} & \cdots & L_{nn} \end{pmatrix} \mapsto \begin{pmatrix} x_1 \in \mathbb{P}^1 \\ \vdots \\ x_{n-1} \in \mathbb{P}^{n-1} \end{pmatrix}. \quad (3.31)$$

3.2.6 Experiments

This section validates our LieBN on nine invariant metrics across the SPD, rotation, and correlation matrices. More details on data sets and experimental settings are provided in Secs. A.1 and A.3.1.

3.2.6.1 Experiments of LieBN on the SPD Manifold

Note that our LieBN layers are architecture-agnostic and can be applied to any existing SPD neural network. Following the previous work [106, 31, 123], we focus on two network architectures: (1) SPDNet [106] for drone recognition on the Radar data set [31], and human action recognition on the HDM05 [153] and FPHA [80] data sets; (2) TSMNet [123] for EEG classification on the Hinss2021 data set [101]. In the EEG application, TSMNet is endowed with *SPD Domain-Specific Momentum Batch Normalization (SPDDSMBN)* (denoted TSMNet+SPDDSMBN) [123], which is a domain adaptation version of Kobler et al. [124]. For a fair comparison, we also implement a domain-specific momentum LieBN, referred to as DSMLieBN. The backbone network architectures are represented as $\{d_0, d_1, \dots, d_L\}$, where the dimension of the parameter in the i -th BiMap layer is $d_i \times d_{i-1}$. As (α, β) only affects variance calculation throughout LieBN, we simply set $(\alpha, \beta) = (1, 0)$ and only tune the deformation factor θ . For each family of LieBN or DSMLieBN, we report two representatives: the standard one induced from the standard metric ($\theta = 1$), and the one induced from the deformed metric with proper θ . *If the standard one is already saturated, we only report the result of that standard variant.*

Application to SPDNet. As SPDNet is a classical SPD network, we apply our LieBN to SPDNet on the Radar, HDM05, and FPHA data sets. Additionally, we compare our method with SPDNetBN, which applies the SPDBN in Eqs. (3.2) and (3.3) to SPDNet. Following Brooks et al. [31], we use the architectures of $\{20, 16, 8\}$, $\{93, 30\}$, and $\{63, 33\}$ for the Radar, HDM05, and FPHA data sets, respectively. The 10-fold av-

3.2. Lie Group Batch Normalization

(a) Radar data set.

Acc	SPDNet	SPDNetBN	SPDNetLieBN				
			$\theta = 1$			Best θ	
			AIM-(1)	LEM-(1)	LCM-(1)	CRIM-(1)	LCM-(-0.5)
Fit time (s)	0.98	1.56	1.62	1.28	1.11	1.82	1.43
Mean $\pm$ STD	93.25 $\pm$ 1.10	94.85 $\pm$ 0.99	95.47$\pm$0.90	94.89 $\pm$ 1.04	93.52 $\pm$ 1.07	94.35 $\pm$ 0.68	94.80 $\pm$ 0.71
Max	94.4	96.13	96.27	96.8	95.2	95.6	95.73

(b) HDM05 data set.

Acc	SPDNet	SPDNetBN	SPDNetLieBN					
			$\theta = 1$			Best θ		
			AIM-(1)	LEM-(1)	LCM-(1)	CRIM-(1)	AIM-(1.5)	LCM-(0.5)
Fit time (s)	0.57	0.97	1.14	0.87	0.66	1.37	1.46	1.01
Mean $\pm$ STD	59.13 $\pm$ 0.67	66.72 $\pm$ 0.52	67.79 $\pm$ 0.65	65.05 $\pm$ 0.63	66.68 $\pm$ 0.71	63.25 $\pm$ 0.88	68.16 $\pm$ 0.68	70.84$\pm$0.92
Max	60.34	67.66	68.75	66.05	68.52	64.94	69.25	72.27

(c) FPFA data set.

Acc	SPDNet	SPDNetBN	SPDNetLieBN					
			$\theta = 1$			Best θ		
			AIM-(1)	LEM-(1)	LCM-(1)	CRIM-(1)	AIM-(1.5)	LCM-(0.5)
Fit time (s)	0.32	0.62	0.8	0.55	0.39	0.92	1.03	0.65
Mean $\pm$ STD	85.59 $\pm$ 0.72	89.33 $\pm$ 0.49	89.70 $\pm$ 0.51	86.56 $\pm$ 0.79	77.64 $\pm$ 1.00	84.65 $\pm$ 1.20	90.39$\pm$0.66	86.33 $\pm$ 0.43
Max	86	90.17	90.5	87.83	79	86.67	92.17	87

Table 3.9: 10-fold average results of SPDNet with and without SPDBN or LieBN on the Radar, HDM05, and FPFA data sets. If the LieBN under the standard metric ($\theta = 1$) is not saturated, the rightmost columns report the deformed LieBN. The best results are **bold**.

erage results, including the average training time (s/epoch), are summarized in Tab. 3.9. We have three key observations regarding the choice of metrics, deformation, and training efficiency.

- **The choice of metrics.** The metric that yields the most effective LieBN layer differs for each data set. Specifically, the optimal LieBN layers on these three data sets are the ones induced by AIM-(1), LCM-(0.5), and AIM-(1.5), respectively, **which improve the performance of SPDNet by 2.22, 11.71, and 4.80 percentage points**. Additionally, although the LCM-based LieBN performs worse than other LieBN variants on the Radar and FPFA data sets, it exhibits the best performance on the HDM05 data set. These observations demonstrate the value of a unified normalization framework that can be instantiated under multiple admissible SPD geometries.
- **The effect of deformation.** Deformation patterns also vary across data sets. Firstly, the standard AIM and CRIM are already saturated on the Radar data set. Secondly, the appropriate deformation θ can further enhance the performance of LieBN. Notably, even though the LieBNs induced by LCM-(1) and CRIM-(1)

(a) Inter-session classification				(b) Inter-subject classification			
Method		Fit Time	Mean $\pm$ STD	Method		Fit Time	Mean $\pm$ STD
SPDDDSMBN		0.16	54.12 $\pm$ 9.87	SPDDDSMBN		7.74	50.10 $\pm$ 8.08
DSMLieBN	AIM-(1)	0.16	55.10$\pm$7.61	DSMLieBN	AIM-(1)	6.94	50.04 $\pm$ 8.01
	LEM-(1)	0.13	54.95 $\pm$ 10.09		LEM-(1)	4.71	50.95 $\pm$ 6.40
	LCM-(1)	0.10	51.54 $\pm$ 6.88		LCM-(1)	3.59	51.86 $\pm$ 4.53
	CRIM-(1)	0.29	51.86 $\pm$ 9.21		CRIM-(1)	16.35	50.71 $\pm$ 8.1
	LCM-(0.5)	0.15	53.11 $\pm$ 5.65		CRIM-(1.5)	19.51	51.34 $\pm$ 5.82
						8.71	53.97$\pm$8.78

Table 3.10: Cross-validation results of TSMNet with SPDDDSMBN and DSMLieBN on the Hinss data set. If the DSMLieBN under the standard metric ($\theta = 1$) is not saturated, the bottom rows report deformed DSMLieBN.

impede the learning of SPDNet on the FPHA data set, they can improve the performance under an appropriate deformation θ . These findings highlight the efficacy of the deforming geometry on the SPD manifold.

- **Efficiency.** Although our LieBN involves additional computations on variance compared with SPDNetBN, our LieBN achieves comparable or even better efficiency than SPDNetBN. In particular, the LieBN induced by standard LEM or LCM exhibits better efficiency than SPDNetBN. Even with deformation, the LCM-based LieBN is still comparable with SPDNetBN in terms of efficiency. This phenomenon could be attributed to the fast and simple computation of LCM and LEM.

Application to EEG classification. We apply our method to TSMNet under two scenarios, inter-session and inter-subject. Following Kobler et al. [123], we adopt the architecture of $\{40, 20\}$. Compared to SPDDDSMBN, DSMLieBN-AIM obtains the highest average scores of 55.10% and 53.97% in these two scenarios, **outperforming SPDDDSMBN by 0.98 and 3.87 percentage points, respectively**. In the inter-subject scenario, the efficiency advantage of our LieBN over SPDDDSMBN is more evident. Specifically, both the LEM- and LCM-based DSMLieBN achieve similar or better performance compared to SPDDDSMBN, while requiring considerably less training time. For example, DSMLieBN-LCM-(1) achieves better results with only half the training time of SPDDDSMBN on inter-subject tasks. Interestingly, under the standard AIM, the sole difference between SPDDDSMBN and our DSMLieBN is the way of centering and biasing. SPDDDSMBN applies the matrix inverse square root and matrix square root to fulfill centering and biasing, while AIM-induced LieBN uses more efficient Cholesky decomposition. As such, the DSMLieBN induced by the standard AIM is more efficient than SPDDDSMBN, particularly on the inter-subject task. On the other hand,

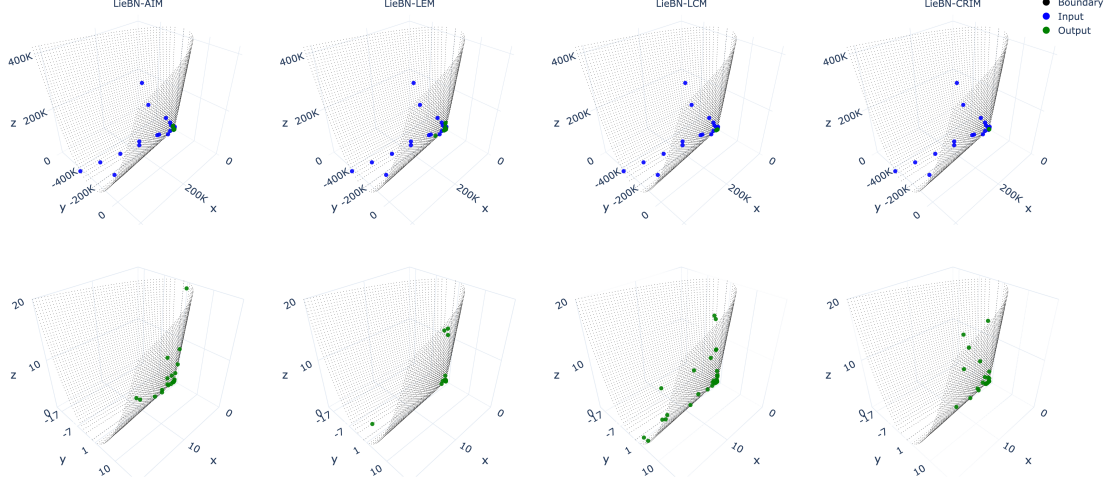


Figure 3.3: Visualization of input and output 30×30 SPD matrices in LieBN using 2×2 Riemannian t-SNE embeddings. The first row shows the input and output under different metrics. Due to the significant difference in magnitude between the t-SNE embeddings of LieBN’s input and output, the second row separately visualizes the LieBN output (at a smaller scale).

the CRIM-based LieBN shows less efficiency, due to the relatively complex Riemannian computation of this metric.

Visualization. We randomly select 50 samples and visualize the input and output of LieBN on the HDM05 data set. Using Riemannian t-SNE [65], we map the 30×30 SPD matrices to 2×2 low-dimensional representations. As shown in Fig. 3.3, LieBN effectively normalizes the data distribution. Specifically, the input t-SNE embeddings are largely scattered and their elements can reach up to 400K, whereas those of the output embeddings are mostly constrained within 20.

3.2.6.2 Experiments of LieBN on Rotation Matrices

This subsection implements our LieBN on the special orthogonal groups, *i.e.*, $SO(n)$, also known as rotation matrices. As the Riemannian metric on $SO(n)$ is bi-invariant, there are two instantiations of our LieBN on this group: LieBN-Left based on the left translation and LieBN-Right based on the right translation. We apply our LieBN to the classic LieNet backbone [107], where the latent space is the special orthogonal group. Following Huang et al. [107], we use three action recognition data sets, the G3D [23], HDM05 [153], and NTU60 [178] data sets. We denote the LieNet models

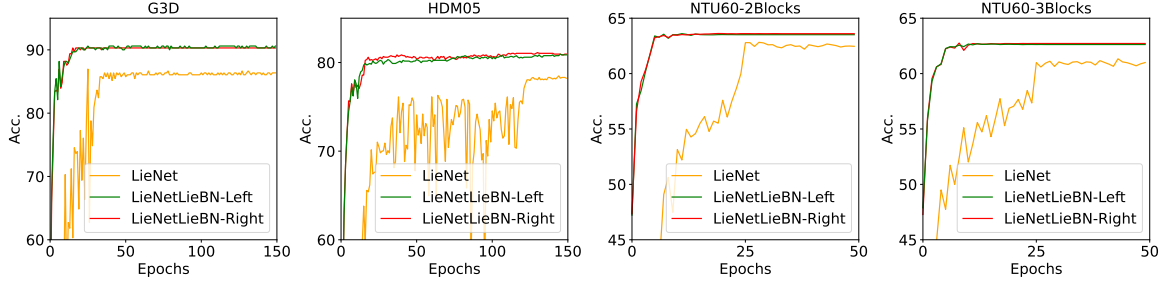


Figure 3.4: Test accuracy curves corresponding to Tab. 3.11.

Method	G3D		HDM05		NTU60	
	Mean $\pm$ STD	Max	Mean $\pm$ STD	Max	2Blocks	3Blocks
LieNet	87.91 $\pm$ 0.90	89.73	76.92 $\pm$ 1.27	79.11	62.4	60.91
LieNetLieBN-Left	88.88$\pm$1.62	90.67	78.89 $\pm$ 1.07	80.88	63.51	62.62
LieNetLieBN-Right	88.12 $\pm$ 1.12	90.3	79.39$\pm$1.13	80.67	63.6	62.72

Table 3.11: Results of LieNet with or without rotation LieBN.

with our LieBN-Left and LieBN-Right as LieNetLieBN-Left and LieNetLieBN-Right, respectively.

Results. We conduct 10-fold experiments on the G3D and HDM05 data sets under the suggested 3Blocks⁵ and 2Blocks architectures, respectively. On the NTU60 data set, we validate LieBN under the 2Blocks and 3Blocks settings. The results are presented in Tab. 3.11. Due to differences in software, our reimplemented LieNet (in PyTorch) performs slightly differently from the results reported by Huang et al. [107] (in MATLAB). However, we still observe a clear improvement when applying our LieBN to the vanilla LieNet backbone. Additionally, LieBN-Right performs slightly better than LieBN-Left. Although the effects of left and right translations on the sample statistics under the bi-invariant metric are identical, their transformations on each sample differ, as illustrated in Fig. 3.1. This difference could slightly affect the network performance. The specific optimal choice of left or right translations depends on the data set’s characteristics.

Training dynamics. Fig. 3.4 presents the test accuracy curves. We have the following additional observations, which can be attributed to the mitigated covariate shift by our LieBN, as our LieBN can effectively normalize the sample statistics.

- **Accelerated convergence.** LieBN significantly accelerates the convergence of LieNet. Specifically, on the NTU60 data set, the largest data set involved, LieNet with LieBN converges by the 5th epoch, whereas the vanilla LieNet does not

⁵Each block consists of a RotMap layer followed by a RotPooling layer. For more details, please refer to Huang et al. [107].

3.3. Gyrogroup Batch Normalization

Data Set	SPDNet	SPDNetLieBN-Cor			
		ECM	LECM	OLM	LSM
HDM05	59.13±0.67	65.37 ± 1.07	61.35 ± 0.34	60.33 ± 0.12	60.00 ± 0.27
FPHA	85.59±0.72	87.20 ± 0.12	87.03 ± 0.32	86.80 ± 0.12	86.77 ± 0.29

Table 3.12: Results of SPDNet with or without correlation LieBN under different invariant metrics.

converge until the 25th epoch. A similar phenomenon can also be observed on the HDM05 data set.

- **More stable performance.** LieBN enhances the stability of network training. In particular, on the HDM05 and G3D data sets, the initial training fluctuations are greatly mitigated by our LieBN.

3.2.6.3 Experiments of LieBN on Correlation Matrices

We apply our correlation LieBN (LieBN-Cor) to SPD networks. Our experiments focus on the SPDNet backbone using the FPHA and HDM05 data sets. LieBN-Cor is applied before the final classification layer. Specifically, SPD features are first activated by the power function, then mapped into correlation matrices via $\text{Cor}(\cdot)$, and finally processed by LieBN-Cor.

Results. The 5-fold average results are presented in Tab. 3.12. Although LieBN-Cor is not specifically designed for SPD networks, it still improves SPDNet’s performance, demonstrating its effectiveness. Among the four invariant metrics, ECM achieves the best performance. LieBN-SPD outperforms LieBN-Cor when applied to SPDNet, as expected because SPDNet is tailored to SPD matrices. However, this does not undermine the validity of LieBN-Cor. The consistent improvement over vanilla SPDNet highlights the potential of applying LieBN-Cor to correlation manifolds.

3.3 Gyrogroup Batch Normalization

3.3.1 Introduction

Although LieBN can normalize sample statistics, many important geometries in machine learning do not admit a Lie group structure. As a result, existing methods still lack a principled solution for Riemannian normalization. Recently, gyro-structures have emerged as effective tools for building Riemannian networks across various geometries,

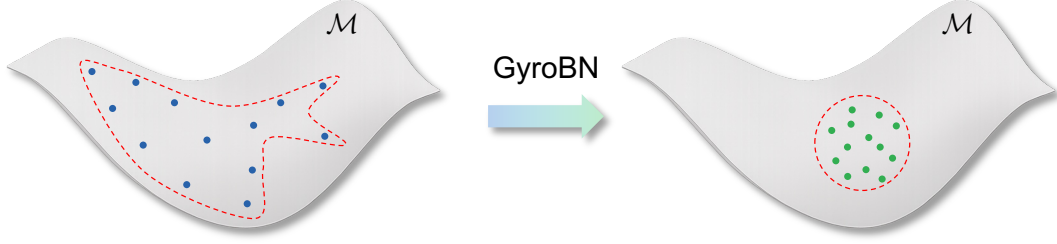


Figure 3.5: Illustration of GyroBN on manifold-valued data. Blue points, green points, and the red dashed curves indicate the input samples, normalized outputs, and data distributions, respectively.

Method	Controllable Statistics	Applied Geometries	Incorporated by GyroBN
SPDBN [31]	M	SPD manifolds under AIM	✓
SPDBN [124]	M+V	SPD manifolds under AIM	✓
SPDDSMBN [123]	M+V	SPD manifolds under AIM	✓
ManifoldNorm [34, Algs. 1–2]	N/A	Riemannian homogeneous spaces	✗
ManifoldNorm [34, Algs. 3–4]	M+V	Matrix Lie groups under the distance $d(X, Y) = \ \log(X^{-1}Y)\ $	✓
RBN [142, Alg. 2]	N/A	Geodesically complete manifolds	✗
LieBN (Sec. 3.2)	M+V	Lie groups under invariant metrics	✓
GyroBN	M+V	Pseudo-reductive gyrogroups with gyroisometric gyrations	N/A

Table 3.13: Comparison of previous RBN methods with GyroBN, where M and V denote the sample mean and variance.

including SPD [157], Grassmannian [157], hyperbolic [76], and spherical manifolds [182]. They naturally extend Euclidean vector structures while encompassing Lie groups and non-group geometries. For instance, the Grassmannian, hyperbolic, and spherical manifolds do not form Lie groups but instead form gyrogroups.

Based on the analysis above, this part of the thesis first introduces the *pseudo-reductive gyrogroup*, a relaxation of the classical gyrogroup that provides a broader algebraic foundation for Riemannian normalization. Building on this structure, we develop GyroBN, a general RBN framework on pseudo-reductive gyrogroups, as illustrated in Fig. 3.5. We employ gyrosubtraction, gyroaddition, and scalar gyromultiplication to generalize the centering (vector subtraction), biasing (vector addition), and scaling (scalar multiplication) in Euclidean BN to curved manifolds in a principled manner. We clarify why centering and biasing in GyroBN rely on *left* gyroaddition, rather than other candidates, such as right gyroaddition or gyrocoaddition [200, Def. 2.9]. We show that when gyrations are *gyroisometries*, GyroBN enjoys theoretical control over sample statistics. These conditions are satisfied by all known gyrogroups in machine learn-

ing, providing a principled and unified normalization mechanism. Moreover, several existing RBN methods arise as special cases of GyroBN, including LieBN and various SPD-based variants, as summarized in Tab. 3.13.

Beyond the LieBN instantiations in Sec. 3.2.5, we instantiate GyroBN on seven representative geometries: the Grassmannian [19], five CCSs [76, 131, 13], and the full-rank correlation manifold [195]. For the Grassmannian, we propose an efficient implementation. For CCSs, we cover five models: Poincaré ball, hyperboloid, Beltrami–Klein, sphere, and projected hypersphere. To enable these instantiations, we refine the projected hypersphere structure [13], derive closed-form gyro-structures for the hyperboloid and sphere, and develop the Riemannian structure of the Beltrami–Klein model. For the correlation manifold, we demonstrate that its gyro-structure can be defined row-wise on its Cholesky factor. We also provide a PyTorch-compatible toolbox [169] with drop-in GyroBN layers, illustrated in Fig. 3.6. Experiments on networks over these seven geometries validate the effectiveness of our framework.

In summary, the main contributions of this part are

- Theoretical foundation: pseudo-reductive gyrogroups as a relaxation of classical gyrogroups;
- General framework: GyroBN as a plug-and-play normalization mechanism, with pseudo-reduction and gyroisometric gyrations ensuring theoretical control of batch statistics;
- Geometric insights: refined projected hypersphere gyro-structure, closed-form hyperboloid and sphere gyro-structures, Riemannian structure of the Beltrami–Klein model, and row-wise correlation manifold gyro-structure;
- Practical instantiations: implementations on the Grassmannian, five CCSs, and the correlation manifold with extensive experiments.⁶

Outline. Sec. 3.3.2 introduces pseudo-reductive gyrogroups and analyzes their theoretical properties, while Sec. 3.3.3 develops the GyroBN framework. Sec. 3.3.4 shows that prior RBN methods are special cases and instantiates GyroBN on seven representative geometries. Sec. 3.3.5 reports experiments that validate GyroBN across these geometries. Proofs are deferred to Sec. B.3.

```

from GyroBN import *
from GyroBN.Geometry import *

# ==== Grassmannian ====
manifold = GrassmannianGyro(n=50, p=10)
X_gr = manifold.random_normal(30, 50, 10)
gybn_gr = GyroBNGr(shape=[50, 10])
out_gr = gybn_gr(X_gr)

# ==== Five CCSs ====
models = [
    ("Poincare", Stereographic(K=-1.0)),
    ("Hyperboloid", Hyperboloid(K=-1.0)),
    ("Klein", Klein(K=-1.0)),
    ("Sphere", Sphere(K= 1.0)),
    ("ProjSphere", Stereographic(K= 1.0)),
]
for name, manifold in models:
    X_ccs = manifold.random_normal(30, 16)
    gybn_ccs = GyroBNCCS(shape=[16], model=name, K=manifold.K)
    out_ccs = gybn_ccs(X_ccs)

# ==== Full-Rank Correlation ====
manifold = CorPolyHyperbolicCholeskyMetric(n=10)
X_cor = manifold.random(30, 10, 10)
gybn_cor = GyroBNCor(shape=[10, 10])
out_cor = gybn_cor(X_cor)

```

Figure 3.6: Minimal examples of applying GyroBN.

3.3.2 Pseudo-Reductive Gyrogroups

Given a gyrogroup $(G, \oplus)$, the left gyrotranslation by $x \in G$ is defined as

$$L_x : G \rightarrow G, \quad L_x(y) = x \oplus y, \quad \forall y \in G. \quad (3.32)$$

If any gyrotranslation is a gyroisometry, we can use gyrotranslation to center manifold-valued samples for the normalization layer. Nguyen and Yang [159] shows that any left gyrotranslation on the SPD and Grassmannian manifolds is a gyroisometry. However, the proof relies on the left cancellation law of gyrogroups, which does not hold for *non-reductive gyrogroups*, such as the Grassmannian. Here, non-reductive gyrogroups refer to gyro-like groupoids that satisfy the first three gyrogroup axioms but fail the

⁶The code is available at <https://github.com/GitZH-Chen/GyroBN.git>.

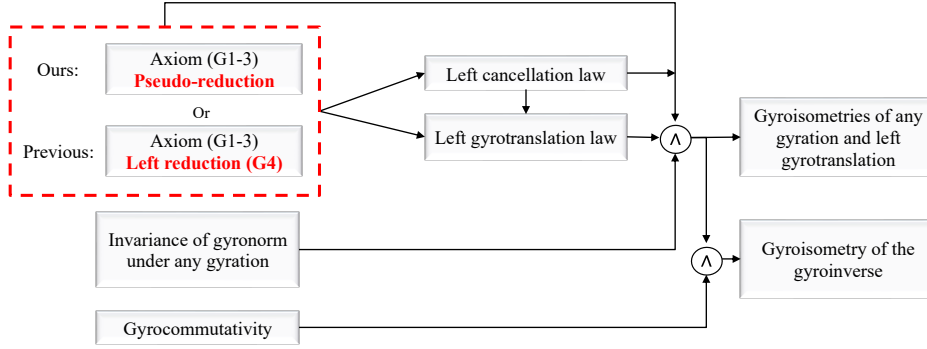


Figure 3.7: A conceptual comparison of the derivation logic in our work with that in previous work [159], where the left gyrotranslation law is presented in Thm. 190. The previous work proves the results on the SPD and Grassmannian manifolds in a case-by-case manner. In contrast, we relax the left reduction into pseudo-reduction and give a general analysis. Our framework also corrects the proof for the Grassmannian cases.

last axiom in Thm. 51, namely the left reduction law (G4), $\text{gyr}[x, y] = \text{gyr}[x \oplus y, y]$. Therefore, the proof is not generally valid for the Grassmannian. We propose an intermediate structure, referred to as *pseudo-reductive gyrogroups*, which supports the left cancellation law and, therefore, the gyrosometry of gyrotranslation. This structure forms the algebraic foundation for building the normalization layer. As illustrated in Fig. 3.7, our derivation extends the case-by-case approach by Nguyen and Yang [159] to general pseudo-reductive gyrogroups.

3.3.2.1 From Gyrogroups to Pseudo-Reductive Gyrogroups

Definition 73 (Pseudo-reductive gyrogroups). A groupoid $(G, \oplus)$ is a pseudo-reductive gyrogroup if it satisfies the axioms (G1), (G2), (G3) and the following pseudo-reductive law:

$$\text{gyr}[a, x] = \text{id}, \text{ for any left inverse } a \text{ of } x \text{ in } G, \quad (3.33)$$

where id is the identity map.

Eq. (3.33) can be intuitively viewed as an intermediate between reduction and non-reduction. For gyrogroups, Eq. (3.33) can be directly obtained from left gyroassociativity (G3) and reduction (G4) [200, Thm. 2.10, item 3]. However, there is no theoretical guarantee that Eq. (3.33) holds for non-reductive gyrogroups. Therefore, we name Eq. (3.33) pseudo-reduction. Nevertheless, for the specific non-reductive Grassmannian, it is indeed pseudo-reductive.

Proposition 74. $\Downarrow$ $\text{Gr}(p, n)$ and $\widetilde{\text{Gr}}(p, n)$ are pseudo-reductive gyrocommutative gyrogroups.

Our pseudo-reductive gyrogroup naturally generalizes the vanilla gyrogroup, as it shares most of the basic properties of gyrogroups [200, Thms. 2.10–2.11].

Theorem 75 (First pseudo-reductive gyrogroup properties). $\Downarrow$ Let $(G, \oplus)$ be a pseudo-reductive gyrogroup. For any elements $x, y, z, a \in G$, we have:

- (1) If $x \oplus y = x \oplus z$, then $y = z$ (General Left Cancellation law; see Case (8) below).
- (2) $\text{gyr}[e, x] = \text{id}$ for any left identity e in G .
- (3) $\text{gyr}[a, x] = \text{id}$ for any left inverse a of x in G .
- (4) There is a left identity that is a right identity.
- (5) There is only one left identity.
- (6) Every left inverse is a right inverse.
- (7) There is only one left inverse, $\ominus x$, of x , and $\ominus(\ominus x) = x$.
- (8) The left cancellation law: $\ominus x \oplus (x \oplus y) = y$.
- (9) The gyrator identity: $\text{gyr}[x, y]a = \ominus(x \oplus y) \oplus \{x \oplus (y \oplus a)\}$.
- (10) $\text{gyr}[x, y]e = e$.
- (11) $\text{gyr}[x, y](\ominus a) = \ominus \text{gyr}[x, y]a$.
- (12) $\text{gyr}[x, e] = \text{id}$.
- (13) The gyrosum inversion law: $\ominus(x \oplus y) = \text{gyr}[x, y](\ominus y \oplus \ominus x)$.

Remark 76. In non-reductive gyrogroups, the identities in Cases (2) and (3) are not guaranteed to hold. Consequently, any property relying on them, such as Case (4) and those from Case (6) to Case (10), is also not guaranteed to hold. The absence of these basic properties undermines the rationality of non-reductive gyrogroups. In contrast, our pseudo-reductive gyrogroups preserve most of the fundamental properties of gyrogroups.

3.3.2.2 Isometries over Pseudo-Reductive Gyrogroups

The gyro-structure in the following is assumed to be defined as Eq. (2.77)–Eq. (2.83). We first clarify that the Riemannian distance agrees with the gyrodistance, and the Riemannian isometry agrees with the gyroisometry. These justify gyrodistance and gyroisometry for gyrospaces over manifolds.

Lemma 77 (Distances). $\downarrow$ *Given a pseudo-reductive gyrogroup $(\mathcal{M}, \oplus)$, we have*

$$d(x, y) = \|\text{Log}_x(y)\|_x = \|\ominus x \oplus y\|_{\text{gyr}} = d_{\text{gyr}}(x, y), \quad \forall x, y \in \mathcal{M}, \quad (3.34)$$

where d denotes the geodesic distance.⁷

Lemma 78 (Isometries). $\downarrow$ *Let $(\mathcal{M}, \oplus)$ and $(\widetilde{\mathcal{M}}, \widetilde{\oplus})$ be two pseudo-reductive gyrogroups. Their gyro identity elements are $e \in \mathcal{M}$ and $\widetilde{e} \in \widetilde{\mathcal{M}}$, respectively. If $\phi : \mathcal{M} \rightarrow \widetilde{\mathcal{M}}$ is a Riemannian isometry with $\widetilde{e} = \phi(e)$, then the following hold.*

(1) *The Riemannian isometry is a gyroisometry:*

$$d_{\text{gyr}}(x, y) = \widetilde{d}_{\text{gyr}}(\phi(x), \phi(y)), \quad (3.35)$$

where d_{gyr} and $\widetilde{d}_{\text{gyr}}$ are the gyrodistances over $\mathcal{M}$ and $\widetilde{\mathcal{M}}$, respectively.

(2) *If the gyroinverse, gyration, or left gyrotranslation over $\mathcal{M}$ is a gyroisometry, its counterpart over $\widetilde{\mathcal{M}}$ is also a gyroisometry.*

Thm. 77 implies that, as long as the gyro-structure is defined by Eq. (2.77)–Eq. (2.83), the gyrodistance coincides with the geodesic distance. Unless otherwise specified, we shall not distinguish between the two and uniformly denote them by $d(\cdot, \cdot)$. Besides, the second result in Thm. 78 is particularly useful, as several geometries are isometric, such as the ONB and PP Grassmannian, as well as different models in hyperbolic geometry.

Now, we analyze gyroisometries over pseudo-reductive gyrogroups. The most related property in Thm. 75 is the left cancellation law, one of the key prerequisites for a gyrotranslation to be a gyroisometry. Note that the left cancellation comes from left gyroassociativity and Eq. (3.33) [200, Thm. 2.10, item 9]. Therefore, left cancellation does not generally hold for non-reductive gyrogroups but exists in pseudo-reductive

⁷On Cartan–Hadamard manifolds, the statement holds for all $x, y \in \mathcal{M}$. More generally, the equality requires x, y to lie within a geodesic ball of convexity radius to ensure the well-definedness of the minimizing geodesic and logarithm. In this section, we implicitly assume these conditions are satisfied.

gyrogroups. We first present an if-and-only-if statement about gyroisometry, which will be useful in the following.

Theorem 79. $\llbracket \downarrow \rrbracket$ *Given a pseudo-reductive gyrogroup $(G, \oplus)$, $\text{gyr}[x, y]$ preserves the gyronorm for any $x, y \in G$ if and only if $\text{gyr}[x, y]$ is a gyroisometry for any $x, y \in G$.*

The gyroisometry of any gyration is a prerequisite for other operators to be gyroisometries.

Theorem 80 (Gyroisometries). $\llbracket \downarrow \rrbracket$ *Given a pseudo-reductive gyrogroup $(G, \oplus)$ with any $\text{gyr}[\cdot, \cdot]$ as a gyroisometry, we have the following.*

- (1) *The left gyrotranslation is a gyroisometry.*
- (2) *If $(G, \oplus)$ is gyrocommutative, then any gyroinverse is a gyroisometry.*

Now, we discuss the gyroisometries for the gyro-structures reviewed in Tabs. 2.5, 2.11 and 2.13.

Theorem 81. $\llbracket \downarrow \rrbracket$ *For the pseudo-reductive gyrogroups corresponding to the SPD manifold (under AIM, LEM, and LCM), the ONB and PP Grassmannian, and the stereographic model with $K \leq 0$ (the Poincaré ball for $K < 0$ and Euclidean space for $K = 0$), the gyrodistance coincides with the geodesic distance. Moreover, the gyroinverse, any gyration, and any left gyrotranslation are gyroisometries.*

Credit and sketch of the proof. As Thm. 77 already shows that the gyrodistance coincides with the geodesic distance, it remains to establish the isometries. For the Grassmannian and SPD manifolds, these were proved by Nguyen and Yang [159, Thms. 2.12–2.14 and 2.16–2.18], although their arguments implicitly treated the non-reductive Grassmannian as a gyrogroup by using left cancellation. Our Thms. 74 and 75 confirms that the Grassmannian is pseudo-reductive and does satisfy left cancellation, thereby validating their results. However, we can directly establish these properties from Thms. 79 and 80. The complete proof is given in Sec. B.3.7. $\square$

Remark 82. The remaining constant-curvature manifolds reviewed in Sec. 2.9.5, including the stereographic model with $K > 0$, radius model, and Beltrami–Klein model, also satisfy these properties. Their verifications will be presented in Sec. 3.3.4.

3.3.3 GyroBN on Pseudo-Reductive Gyrogroups

Building on Thm. 81, which establishes that several geometries admit isometric gyrotranslations, we develop RBN in a principled way for general pseudo-reductive gyrogroups, referred to as GyroBN. Throughout, we assume $(\mathcal{M}, \oplus)$ is a pseudo-reductive gyrogroup with the gyro-structure defined by Eq. (2.77)–Eq. (2.83).⁸ As Thm. 77 establishes the equivalence between gyrodistance and geodesic distance, we use the terms “gyromean” and “gyrovariance” interchangeably with their Riemannian counterparts.

3.3.3.1 GyroBN

To generalize the Euclidean BN in Eq. (3.1) to gyrogroups, we first define sample mean, sample variance, centering, biasing, and scaling over gyrogroups. Then, we introduce the GyroBN framework with a theoretical analysis of the ability to normalize sample statistics.

We define the gyromean as the Fréchet mean [74] under gyrodistance:

$$\mu = \text{FM}(\{x_i \in \mathcal{M}\}_{i=1}^N) = \underset{y \in \mathcal{M}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^N d^2(x_i, y). \quad (3.36)$$

The gyrovariance is the corresponding Fréchet variance. By Thm. 77, the gyromean and gyrovariance coincide with the Riemannian mean and variance, *i.e.*, the Fréchet mean and variance under geodesic distance. The existence and local uniqueness of the Fréchet mean are reviewed in Thm. 32.

Easy computation shows that the Euclidean BN operations in Eq. (3.1) have direct gyrogroup counterparts in $\mathbb{R}^n$. Centering corresponds to gyrosubtraction (Eqs. (2.77) and (2.79)), biasing to gyroaddition (Eq. (2.77)), and scaling to scalar gyromultiplication (Eq. (2.78)). Motivated by this, we define a normalization layer over gyrogroups via gyro operations. Given a batch of activations $\{x_i\}_{i=1}^N \subset \mathcal{M}$, the core operations of GyroBN are

$$\forall i \leq N, \quad \tilde{x}_i \leftarrow \overbrace{\beta \oplus}^{\text{Biasing}} \left(\overbrace{\frac{s}{\sqrt{v^2 + \epsilon}} \odot}^{\text{Scaling}} \left(\overbrace{\ominus \mu \oplus x_i}^{\text{Centering}} \right) \right), \quad (3.37)$$

where $\mu \in \mathcal{M}$ and v^2 are gyromean and gyrovariance, $\beta \in \mathcal{M}$ is the bias parameter, $s \in \mathbb{R}$ is the scaling parameter, and ϵ is a small value for numerical stability. The following theorem shows that Eq. (3.37) can normalize manifold-valued data.

⁸In GyroBN, $\odot$ is not required to satisfy the axioms of a gyrovector space (Thm. 53).

Algorithm 2: Gyrogroup Batch Normalization (GyroBN)

Require : batch of activations $\{x_i\}_{i=1}^N \subset \mathcal{M}$, small positive constant ϵ , and momentum $\eta \in [0, 1]$, running mean μ_r , running variance v_r^2 , bias parameter $\beta \in \mathcal{M}$, scaling parameter $s \in \mathbb{R}$.
Return : normalized batch $\{\tilde{x}_i\}_{i=1}^N \subset \mathcal{M}$
if *training* **then**
 Compute batch mean μ_b and variance v_b^2 of $\{x_i\}_{i=1}^N$;
 Update running statistics $\mu_r = \text{Bar}_\eta(\mu_b, \mu_r)$, and $v_r^2 = \eta v_b^2 + (1 - \eta)v_r^2$;
 $(\mu, v^2) = (\mu_b, v_b^2)$ **if** *training* **else** (μ_r, v_r^2)
 $\forall i \leq N, \tilde{x}_i = \beta \oplus \left(\frac{s}{\sqrt{v^2 + \epsilon}} \odot (\ominus \mu \oplus x_i) \right)$

Theorem 83 (Homogeneity). $\downarrow$ Let $(\mathcal{M}, \oplus)$ be a pseudo-reductive gyrogroup in which every gyration $\text{gyr}[\cdot, \cdot]$ is a gyroisometry. For N samples $\{x_i\}_{i=1}^N \subset \mathcal{M}$ and any $t \in \mathbb{R}$, we have

$$\text{Homogeneity of gyromean: } \text{FM}(\{\beta \oplus x_i\}_{i=1}^N) = \beta \oplus \text{FM}(\{x_i\}_{i=1}^N), \quad \forall \beta \in \mathcal{M}, \quad (3.38)$$

$$\text{Homogeneity of dispersion from } e: \frac{1}{N} \sum_{i=1}^N d^2(t \odot x_i, e) = \frac{t^2}{N} \sum_{i=1}^N d^2(x_i, e), \quad (3.39)$$

The most important property of the Euclidean BN [109] lies in its ability to normalize the sample mean and variance. Thm. 83 shows that the formulation in Eq. (3.37) enjoys the same property: homogeneity of the gyromean guarantees that centering and biasing shift the gyromean, while homogeneity of the dispersion from e ensures that scaling controls the sample variance. As a result, GyroBN provides a theoretical guarantee of normalization on any pseudo-reductive gyrogroup with isometric gyrations. Moreover, since the gyromean and gyrovariance coincide with their Riemannian counterparts, GyroBN also normalizes Riemannian statistics.

To finalize GyroBN, we define the running mean updates over gyrogroups as the binary barycenter based on gyrodistance:

$$\text{Bar}_\eta(x_1, x_2) = \operatorname{argmin}_{y \in \mathcal{M}} \left(\eta d^2(x_1, y) + (1 - \eta) d^2(x_2, y) \right), \quad \eta \in [0, 1], \quad (3.40)$$

which can be calculated by the geodesic. With these ingredients, the general framework for GyroBN is presented in Alg. 2. In particular, it recovers the classic Euclidean

BN [109] when $\mathcal{M} = \mathbb{R}^n$.

Remark 84. We make the following two remarks with respect to left gyrotranslation.

- **Other candidates.** There are three alternatives to left gyrotranslation. However, they are not necessarily gyroisometries and therefore cannot support the general GyroBN construction without additional isometry assumptions. Specifically, analogous to the left gyrotranslation, the right gyrotranslation is

$$R_x : G \rightarrow G, \quad R_x(y) = y \oplus x, \quad \forall y \in G. \quad (3.41)$$

Along with the gyroaddition, there is the gyrogroup coaddition [200, Def. 2.9]:

$$x \boxplus y = x \oplus \text{gyr}[x, \ominus y]y, \quad \forall x, y \in G. \quad (3.42)$$

Coaddition is symmetric to gyroaddition in many ways; for instance, when gyroaddition is gyrocommutative, coaddition is commutative [200, Thm. 3.3]. However, the right gyrotranslation, as well as the left and right translations by coaddition, are not guaranteed to be gyroisometries. Numerical experiments confirm that these three translations on the hyperbolic Poincaré ball fail to preserve gyrodistance (see `gyrocoadd.py`). A theoretical reason is that they all lack a counterpart of the left gyrotranslation law, which is important for the translation to be a gyroisometry (see Fig. 3.7).

- **Special cases.** For Lie groups with a right-invariant Riemannian metric, right translations are isometries, and GyroBN can then be formulated using them, with Thm. 83 extending directly. In general, however, only left gyrotranslations are guaranteed to be isometries.

3.3.4 Instantiations

As indicated by Thms. 81 and 83, GyroBN can be applied to different geometries with guaranteed normalization of the sample statistics. Once the required operators are specified, Alg. 2 can be used in a plug-and-play manner. We first show that existing RBN methods with control over sample statistics, such as LieBN on Lie groups and AIM-based SPDBNs, are special cases of the framework. We then instantiate GyroBN on seven representative geometries: the Grassmannian, five CCS models (Poincaré ball, projected hypersphere, hyperboloid, sphere, and Beltrami–Klein), and the full-rank

correlation manifold. To support these instantiations, we simplify the Grassmannian operators for efficient computation, refine the gyro-structures on the Poincaré ball and projected hypersphere, establish new gyro-structures for the hyperboloid and sphere, characterize the Beltrami–Klein geometry, and formulate a row-wise realization for the correlation manifold.

3.3.4.1 LieBN as a Special Case

Chakraborty [34, Algs. 3–4] introduced the Riemannian normalization on matrix Lie groups under a specific distance. The extension to general Lie groups, yielding LieBN with theoretical normalization over the Riemannian mean and variance, is presented in Sec. 3.2. This subsection shows that LieBN is a special case of GyroBN.

LieBN is formulated with an invariant metric on a Lie group. Centering and biasing are performed through group translations, while scaling is defined in the tangent space at the identity element. Since every Lie group is automatically a gyrogroup, gyrotranslation reduces exactly to the group translation. Consequently, the centering, biasing, and scaling in LieBN are identical to those in GyroBN. Moreover, as shown in Thm. 77, the mean, variance, and running mean update defined via the geodesic distance in LieBN are equivalent to their counterparts based on gyrodistance in GyroBN. Therefore, LieBN is a special case of GyroBN. The LieBN instantiations presented in Sec. 3.2.5 are therefore incorporated by GyroBN.

3.3.4.2 AIM-Based SPDBNs as Special Cases

As reviewed in Sec. 3.2.3.2, several RBNs on the SPD manifold were developed based on AIM [31, 124, 123]. These SPD normalization methods can be expressed in the following unified form:

$$\text{Normalization: } \forall i \leq N, \quad \tilde{P}_i \leftarrow B^{\frac{1}{2}} \left(M^{-\frac{1}{2}} P_i M^{-\frac{1}{2}} \right)^{\frac{s}{\sqrt{v^2 + \epsilon}}} B^{\frac{1}{2}}, \quad (3.43)$$

where M and v^2 are the Riemannian mean and variance. The running mean is updated by the binary barycenter under the geodesic distance.

As gyrodistance is identical to the geodesic distance, the gyromean, gyrovariance, and running mean updates are identical to the Riemannian ones. Using the AIM gyro-operations in Eq. (2.95), Eq. (3.43) is exactly the specific implementation of Eq. (3.37) under the AIM-based gyrogroup on the SPD manifold. Therefore, the SPDBNs developed by Brooks et al. [31], Kobler et al. [124, 123] are also special cases of GyroBN.

Remark 85. Brooks et al. [31] only considers centering and biasing. Kobler et al. [124] uses a running mean for centering during training. Kobler et al. [123] uses different momentum parameters to update running statistics for training and testing, along with multi-channel mechanisms for domain adaptation. Nevertheless, all of them are based on Eq. (3.43). Therefore, tricks such as multi-channel and separate momentum can also be applied to GyroBN. This is what we mean by claiming that GyroBN incorporates their approaches.

3.3.4.3 Grassmannian Manifold

We focus on the ONB perspective. Let $U, V \in \text{Gr}(p, n)$, $t \in \mathbb{R}$, and $\Delta \in T_U \text{Gr}(p, n)$. Using the Grassmannian operators reviewed in Tabs. 2.10 and 2.11, we instantiate GyroBN on the ONB Grassmannian.

Instantiation. Building on these operators, we now implement the ONB Grassmannian GyroBN. Given a batch of activations $\{U_{1 \dots N}\}$, the three core steps of GyroBN are

$$\text{Centering to the identity } I_{p,n}: U_i^1 = \exp \left(-[\overline{MM^\top}, \tilde{I}_{p,n}] \right) U_i, \quad (3.44)$$

$$\text{Scaling the dispersion from } I_{p,n}: U_i^2 = \exp \left(\frac{s}{\sqrt{v^2 + \epsilon}} [\overline{U_i^1 (U_i^1)^\top}, \tilde{I}_{p,n}] \right) I_{p,n}, \quad (3.45)$$

$$\text{Biasing towards } B \in \mathcal{M}: U_i^3 = \exp \left([\overline{B}, \tilde{I}_{p,n}] \right) U_i^2. \quad (3.46)$$

Here $\overline{(\cdot)} = \widetilde{\text{Log}}_{\tilde{I}_{p,n}}(\cdot)$ is the Riemannian logarithm under the PP Grassmannian, M (resp. v^2) is the Riemannian batch mean (resp. variance), and $\tilde{I}_{p,n} = I_{p,n} I_{p,n}^\top$ is the PP identity. The mean M can be obtained by the Karcher flow [116], with the Riemannian logarithm computed by Bendokat et al. [19, Alg. 5.3].

Efficient computation. The commutators $[\overline{MM^\top}, \tilde{I}_{p,n}]$ and $[\overline{U_i^1 (U_i^1)^\top}, \tilde{I}_{p,n}]$ can be efficiently computed by the following result.

Proposition 86. $\Downarrow$ Given $U = (U_1^\top, U_2^\top)^\top \in \text{Gr}(p, n)$ with $U_1 \in \mathbb{R}^{p \times p}$ and $U_2 \in \mathbb{R}^{(n-p) \times p}$, then

$$[\overline{UU^\top}, \tilde{I}_{p,n}] = \begin{pmatrix} \mathbf{0}_{p \times p} & -\tilde{U}_2^\top \\ \tilde{U}_2 & \mathbf{0}_{(n-p) \times (n-p)} \end{pmatrix}, \quad (3.47)$$

where $\tilde{U}_2 = U_2 Q \frac{\arcsin(\hat{S})}{\hat{S}} R^\top$ and $U_1^\top \stackrel{\text{SVD}}{=} QSR^\top$. Here S is in ascending order, Q and R are flipped column-wise, and $\hat{S} = \sqrt{I_p - S^2}$.

Remark 87. Two technical issues are worth noting.

- **Cut locus.** The logarithm $\text{Log}_U(V)$ exists only when U and V are not in each other's cut locus [19]. Similarly, gyroaddition and gyromultiplication are not globally defined due to the cut locus [157, Sec. 3.2]. However, Bendokat et al. [19, Alg. 5.3] provides a numerical remedy.
- **PP Grassmannian.** Although our derivation is based on the ONB Grassmannian, GyroBN under the PP Grassmannian can be obtained by mapping data via $\pi^{-1} : \widetilde{\text{Gr}}(p, n) \rightarrow \text{Gr}(p, n)$, normalizing, and mapping back via π . This follows from the isometry $\pi : \text{Gr}(p, n) \rightarrow \widetilde{\text{Gr}}(p, n)$.

3.3.4.4 Stereographic Model

We begin by analyzing its gyro-structure and then instantiate GyroBN.

Stereographic gyrovector space. As reviewed in Sec. 2.9.5, the stereographic model $\mathfrak{st}_K^n$ unifies CCS geometries: the hyperbolic Poincaré ball $\mathbb{P}_K^n$ for $K < 0$, Euclidean space $\mathbb{R}^n$ for $K = 0$, and the spherical projected hypersphere $\mathbb{D}_K^n$ for $K > 0$. For $x, y, z \in \mathfrak{st}_K^n$, $t \in \mathbb{R}$, and $v \in T_x \mathfrak{st}_K^n$, its Riemannian and gyro operators are reviewed in Tabs. 2.13 and 2.14.

For $K < 0$, the Poincaré ball $(\mathbb{P}_K^n, \oplus_K, \odot_K)$ forms a Möbius gyrovector space, as reviewed in Tab. 2.13. For $K > 0$, however, the situation is subtler [13]:

- gyroaddition is well-defined except when $x = \frac{y}{K\|y\|^2}$;
- gyromultiplication is well-defined except when $r \tan^{-1}(\sqrt{K}\|x\|) = \pi/2 + k\pi$ for some $k \in \mathbb{Z}$.

Even if assumed well-defined, it remains unclear whether $(\mathfrak{st}_K^n, \oplus_K, \odot_K)$ with $K > 0$ satisfies the axioms of a gyrovector space, in contrast to the Poincaré ball. Closing this gap is part of our contribution: under the assumption of well-definedness, we show that the stereographic gyro operations coincide with those in Eqs. (2.77) and (2.78) and prove that the stereographic model satisfies all axioms of a gyrovector space.

Proposition 88. [↓] *The stereographic gyroaddition and gyromultiplication coincide with the Riemannian definitions:*

$$x \oplus_K y = \text{Exp}_x(\text{PT}_{\mathbf{0} \rightarrow x}(\text{Log}_{\mathbf{0}}(y))), \quad \forall x, y \in \mathfrak{st}_K^n, \quad (3.48)$$

$$t \odot_K x = \text{Exp}_{\mathbf{0}}(t \text{Log}_{\mathbf{0}}(x)), \quad \forall t \in \mathbb{R}, x \in \mathfrak{st}_K^n. \quad (3.49)$$

Theorem 89. $\Downarrow$ For any $K \in \mathbb{R}$, the stereographic model $(\mathfrak{st}_K^n, \oplus_K)$ satisfies all axioms of a gyrocommutative gyrogroup. When further endowed with gyromultiplication $\odot_K$, it satisfies all axioms of a gyrovector space.

Stereographic GyroBN. Next, we extend Thm. 81 to the stereographic model with arbitrary curvature.

Theorem 90. $\Downarrow$ For the stereographic model, the gyrodistance coincides with the geodesic distance. Moreover, the gyroinverse, any gyration, and any left gyrotranslation are gyroisometries.

Combining Thm. 83 and Thm. 90, GyroBN in the stereographic model is theoretically guaranteed to normalize sample statistics. Practically, implementation only requires substituting the stereographic operators reviewed in Tabs. 2.13 and 2.14 into Alg. 2. For efficient computation, the Poincaré Fréchet mean can be obtained using the algorithm of Lou et al. [142, Alg. 1], while the mean on the sphere is computed via the Karcher flow [116].

3.3.4.5 Radius Model

As reviewed in Sec. 2.9.5, the radius model $\mathcal{M}_K^n$ is another representation of constant-curvature spaces, unifying the hyperboloid $\mathbb{H}_K^n$ for $K < 0$, the sphere $\mathbb{S}_K^n$ for $K > 0$, and the Euclidean space $\mathbb{R}^n$ for $K = 0$. The case $K = 0$ reduces trivially to Euclidean space, so the following analysis of the radius-model gyro-structure considers $K \neq 0$. Although this model has been effective in various applications [38, 45, 15, 166, 99, 119], its gyro-structure has not been formalized. We first analyze the gyro-structure on $\mathcal{M}_K^n$ and then instantiate GyroBN.

Radius gyrovector space. The radius model $\mathcal{M}_K^n$ is isometric to the stereographic model $\mathfrak{st}_K^n$ via stereographic projection fixing the south pole [182]:

$$\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n} : \mathcal{M}_K^n \ni \begin{bmatrix} \xi \in \mathbb{R} \\ x \in \mathbb{R}^n \end{bmatrix} \mapsto \frac{x}{1 + \sqrt{|K|}\xi} \in \mathfrak{st}_K^n, \quad (3.50)$$

$$\pi_{\mathfrak{st}_K^n \rightarrow \mathcal{M}_K^n} : \mathfrak{st}_K^n \ni y \mapsto \begin{bmatrix} \frac{1}{\sqrt{|K|}} \frac{1-K\|y\|^2}{1+K\|y\|^2} \\ \frac{2y}{1+K\|y\|^2} \end{bmatrix} \in \mathcal{M}_K^n. \quad (3.51)$$

The origin in $\mathcal{M}_K^n$ is defined as $\bar{\mathbf{0}} = [\sqrt{1/|K|}, 0, \dots, 0]^\top$, corresponding to $\mathbf{0} \in \mathfrak{st}_K^n$. For $K > 0$, we have $\mathcal{M}_K^n = \mathbb{S}_K^n$ and $\mathfrak{st}_K^n = \mathbb{D}_K^n$. Since Eq. (3.50) is undefined at the

south pole $-\bar{\mathbf{0}}$ when $K > 0$, we use the one-point compactification $\mathbb{D}_K^n \cup \{\infty\}$ with the identification $\pi_{\mathcal{M}_K^n \rightarrow \text{st}_K^n}(-\bar{\mathbf{0}}) = \infty$ [182, Rmk. A.9]. For simplicity, we use $\mathbb{D}_K^n$ and $\mathbb{D}_K^n \cup \{\infty\}$ interchangeably.

Tab. 2.14 reviews the Riemannian operators. We adopt the following curvature-aware functions:

$$\begin{aligned} \sin_K &= \begin{cases} \sin & \text{if } K > 0, \\ \sinh & \text{if } K < 0, \end{cases} & \cos_K &= \begin{cases} \cos & \text{if } K > 0, \\ \cosh & \text{if } K < 0, \end{cases} \\ \langle \cdot, \cdot \rangle_K &= \begin{cases} \langle \cdot, \cdot \rangle & \text{if } K > 0, \\ \langle \cdot, \cdot \rangle_{\mathcal{L}} & \text{if } K < 0. \end{cases} \end{aligned} \quad (3.52)$$

For $v \in T_x \mathcal{M}_K^n$, $\|v\|_K = \sqrt{\langle v, v \rangle_K}$ is the induced Riemannian norm. On the $K < 0$ branch, the ambient Lorentzian form is positive definite only after this tangent-space restriction. Moreover, $(\cdot)_s$ denotes the space vector, and $(\cdot)_t$ denotes the time scalar.

Using the radius-model Riemannian operators reviewed in Tab. 2.14, we define the gyroaddition and gyromultiplication as Eqs. (2.77) and (2.78):

$$x \oplus_K^{\mathcal{M}} y = \text{Exp}_x(\text{PT}_{\bar{\mathbf{0}} \rightarrow x}(\text{Log}_{\bar{\mathbf{0}}}(y))), \quad \forall x, y \in \mathcal{M}_K^n, \quad (3.53)$$

$$t \odot_K^{\mathcal{M}} x = \text{Exp}_{\bar{\mathbf{0}}}(t \text{Log}_{\bar{\mathbf{0}}}(x)), \quad \forall t \in \mathbb{R}, \forall x \in \mathcal{M}_K^n. \quad (3.54)$$

We give the following clarifications regarding the above two gyro operations.

- For hyperbolic geometry ($K < 0$), Eq. (3.53) has been employed in prior work [38, 99]. However, there exists no closed-form expression, which could be more efficient than the composition of Riemannian operators. Besides, the underlying gyro-structure has not been formally discussed.
- For spherical geometry ($K > 0$), the geodesic between antipodal points (x and $-x$) is not unique, making the logarithm and parallel transport along such a geodesic ill-defined. For the sphere, Eq. (3.53) assumes that $x \neq -\bar{\mathbf{0}}$ and $y \neq -\bar{\mathbf{0}}$, while Eq. (3.54) assumes that $x \neq -\bar{\mathbf{0}}$. Like before, we always make these assumptions implicitly.

In the following, we first give the closed-form expressions of Eqs. (3.53) and (3.54). Then, we show that Eqs. (3.53) and (3.54) conform to all the axioms of a gyrovector space.

Proposition 91 (Gyromultiplication and gyroinverse). $\downarrow$ Let $x = [x_t, x_s^\top]^\top$ be a point in $\mathcal{M}_K^n$, where $x_t \in \mathbb{R}$ is the time scalar, and $x_s \in \mathbb{R}^n$ is the spatial part. The gyromultiplication and inverse have closed-form expressions:

$$t \odot_K^{\mathcal{M}} x = \begin{cases} \bar{\mathbf{0}}, & t = 0 \vee x = \bar{\mathbf{0}}, \\ \frac{1}{\sqrt{|K|}} \begin{bmatrix} \cos_K \left(t \cos_K^{-1}(\sqrt{|K|}x_t) \right) \\ \frac{\sin_K \left(t \cos_K^{-1}(\sqrt{|K|}x_t) \right)}{\|x_s\|} x_s \end{bmatrix}, & t \neq 0, \end{cases} \quad (3.55)$$

$$\ominus_K^{\mathcal{M}} x = -1 \odot_K^{\mathcal{M}} x = \begin{bmatrix} x_t \\ -x_s \end{bmatrix}. \quad (3.56)$$

In particular, the gyro identity is $\bar{\mathbf{0}}$. Besides, the following shows that the sphere gyromultiplication is still valid for the singular cases in the projected hypersphere. Assume $K > 0$ and $x \neq \pm \bar{\mathbf{0}}$, and let $\mathfrak{st}_K^n \ni u = \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(x)$ be its stereographic image. For $t \in \mathbb{R}$, set $\theta = \cos^{-1}(\sqrt{K}x_t) \in (0, \pi)$. Then the following are equivalent:

$$(i) \ t \tan^{-1}(\sqrt{K}\|u\|) = \frac{\pi}{2} + k\pi \iff (ii) \ t\theta = (2k+1)\pi, \quad k \in \mathbb{Z}. \quad (3.57)$$

In these singular cases, Eq. (3.55) is still valid, while the stereographic gyromultiplication returns infinity:

$$t \odot_K^{\mathcal{M}} x = -\bar{\mathbf{0}} \in \mathcal{M}_K^n, \quad t \odot_K u = \infty \in \mathfrak{st}_K^n, \quad (3.58)$$

where ∞ denotes the added point in the one-point compactification $\mathbb{D}_K^n \cup \{\infty\}$ ($\mathfrak{st}_K^n = \mathbb{D}_K^n$ for $K > 0$) and $\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(-\bar{\mathbf{0}}) = \infty$.

Proposition 92 (Gyroaddition). $\downarrow$ Let $x = [x_t, x_s^\top]^\top$ and $y = [y_t, y_s^\top]^\top$ be points in $\mathcal{M}_K^n$, where $x_t, y_t \in \mathbb{R}$ are the time scalars, and $x_s, y_s \in \mathbb{R}^n$ are the spatial parts.

Then, the gyroaddition on $\mathcal{M}_K^n$ admits the closed form:

$$x \oplus_K^{\mathcal{M}} y = \begin{cases} x, & y = \bar{\mathbf{0}}, \\ y, & x = \bar{\mathbf{0}}, \\ \begin{bmatrix} \frac{1}{\sqrt{|K|}} \frac{D-KN}{D+KN} \\ \frac{2(A_s x_s + A_y y_s)}{D+KN} \end{bmatrix}, & \text{otherwise.} \end{cases} \quad (3.59)$$

Here, $A_s = ab^2 - 2Kbs_{xy} - Kan_y$ and $A_y = b(a^2 + Kn_x)$, where the following quantities are defined by

$$a = 1 + \sqrt{|K|}x_t, b = 1 + \sqrt{|K|}y_t, n_x = \|x_s\|^2, n_y = \|y_s\|^2, s_{xy} = \langle x_s, y_s \rangle. \quad (3.60)$$

$$D = a^2b^2 - 2Kabs_{xy} + K^2n_xn_y, N = a^2n_y + 2abs_{xy} + b^2n_x. \quad (3.61)$$

Besides, the following shows that the sphere gyroaddition is still valid under the singular cases in the projected hypersphere. Assume $K > 0$ and $x, y \neq \pm \bar{\mathbf{0}}$, and let $u = \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{sl}_K^n}(x)$ and $v = \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{sl}_K^n}(y)$ be the stereographic images. The following statements are equivalent:

- (1) $u = \frac{v}{K\|v\|^2}$ ($v \neq \mathbf{0}$);
- (2) $x_s = y_s$ and $x_t = -y_t$ (same meridian, mirrored across the equator);
- (3) $D = 0$.

In such singular cases, we have $N > 0$ and Eq. (3.59) is still valid, while the stereographic gyroaddition returns infinity:

$$x \oplus_K^{\mathcal{M}} y = -\bar{\mathbf{0}}, \quad u \oplus_K v = \infty, \quad (3.62)$$

where ∞ is the point added in the one-point compactification of $\mathbb{D}_K^n$.

The above two propositions immediately imply that $\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{sl}_K^n}$ preserves the gyro operations.

Corollary 93 (Isomorphism). $\Downarrow$ For the hyperbolic geometry ($K < 0$), the isom-

3.3. Gyrogroup Batch Normalization

etry $\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n} : \mathbb{H}_K^n \rightarrow \mathbb{P}_K^n$ preserves the gyro operations:

$$\begin{aligned} x \oplus_K^{\mathcal{M}} y &= \pi_{\mathfrak{st}_K^n \rightarrow \mathcal{M}_K^n} (\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(x) \oplus_K \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(y)), \quad \forall x, y \in \mathcal{M}_K^n, \\ r \odot_K^{\mathcal{M}} x &= \pi_{\mathfrak{st}_K^n \rightarrow \mathcal{M}_K^n} (r \odot_K \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(x)), \quad \forall r \in \mathbb{R}, \forall x \in \mathcal{M}_K^n. \end{aligned} \quad (3.63)$$

For the spherical geometry ($K > 0$), the isometry $\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n} : \mathbb{S}_K^n \rightarrow \mathbb{D}_K^n \cup \{\infty\}$ preserves the gyro operations:

$$\begin{aligned} x \oplus_K^{\mathcal{M}} y &= \pi_{\mathfrak{st}_K^n \rightarrow \mathcal{M}_K^n} (\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(x) \oplus_K \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(y)), \quad \forall x, y \in \mathcal{M}_K^n / \{-\bar{\mathbf{0}}\}, \\ r \odot_K^{\mathcal{M}} x &= \pi_{\mathfrak{st}_K^n \rightarrow \mathcal{M}_K^n} (r \odot_K \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(x)), \quad \forall r \in \mathbb{R}, \forall x \in \mathcal{M}_K^n / \{-\bar{\mathbf{0}}\}. \end{aligned} \quad (3.64)$$

Remark 94. We provide two clarifications regarding Thm. 93.

- **Compactified projected hypersphere.** Although stereographic operations for $K > 0$ may be undefined in certain cases, they become well-defined on the one-point compactification $\mathbb{D}_K^n \cup \{\infty\}$, where undefined cases correspond to ∞ .
- **Sphere vs. projected hypersphere.** Thm. 93 suggests a numerical advantage of the sphere: gyro-operations are well-defined on $\mathbb{S}_K^n$ at all points except the single south pole $-\bar{\mathbf{0}}$, including cases corresponding to singularities on the projected hypersphere. This broader domain can make computations on $\mathbb{S}_K^n$ more stable.

From the above corollary, it is expected that the operations $\oplus_K^{\mathcal{M}}$ and $\odot_K^{\mathcal{M}}$ also satisfy the axioms of a gyrovector space for both negative and positive curvature K .

Theorem 95 (Radius gyrovector spaces). $\lfloor \downarrow \rfloor (\mathcal{M}_K^n, \oplus_K^{\mathcal{M}})$ forms a gyrocommutative gyrogroup, and $(\mathcal{M}_K^n, \oplus_K^{\mathcal{M}}, \odot_K^{\mathcal{M}})$ forms a gyrovector space.^a

^aFor $K > 0$, we implicitly assume the addition and multiplication are well-defined; whenever they are, all corresponding axioms hold.

Due to the isometry, Thm. 78 implies gyroisometries over $\mathcal{M}_K^n$.

Theorem 96. On the radius model, the gyrodistance is identical to the geodesic distance, whereas the gyroinverse, gyration, and left gyrotranslation are gyroisometries.

Radius GyroBN. Thm. 96 guarantees that GyroBN on the radius model normal-

izes the sample mean and variance. Substituting the radius Riemannian operators from Tab. 2.14 and the gyro operators from Thms. 91 and 92 into Alg. 2 can directly yield the radius GyroBN. For $K < 0$ (hyperboloid), the Fréchet mean can be computed efficiently by Lou et al. [142, Alg. 3]; for $K > 0$ (sphere), we compute the Fréchet mean using the Karcher flow [116].

3.3.4.6 Hyperbolic Beltrami–Klein

The Beltrami–Klein model has recently emerged as a promising alternative to the Poincaré ball for representing hyperbolic geometry [147]. As reviewed in Sec. 2.9.5, the Poincaré ball admits the Möbius gyrovector space and the Beltrami–Klein model admits the Einstein gyrovector space. While prior studies mainly focused on the case $K = -1$ [147], we develop the Beltrami–Klein Riemannian structure under arbitrary negative curvature and relate it to the Einstein gyrospace. This allows us to establish the equivalence between the gyro and Riemannian formulations and, ultimately, to instantiate GyroBN on this model.

Beltrami–Klein Riemannian structure. The standard Einstein gyro operations on the Beltrami–Klein model are reviewed in Tab. 2.13. For $K = -1$, prior work related the Einstein operations to the Riemannian-form gyro operations in Eqs. (2.77) and (2.78) [147, Sec. 4.2]. We extend this equivalence to arbitrary $K < 0$ and derive the corresponding closed-form Beltrami–Klein Riemannian operators. To this end, we first establish the isometry between the Beltrami–Klein and Poincaré models.

Proposition 97 (Beltrami–Klein isometries). [↓] *The following maps are Riemannian isometries between the Beltrami–Klein and Poincaré ball models:*

$$\pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n} : \mathbb{K}_K^n \ni x \mapsto \frac{1}{1 + \sqrt{1 + K \|x\|^2}} x \in \mathbb{P}_K^n, \quad (3.65)$$

$$\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n} : \mathbb{P}_K^n \ni x \mapsto \frac{2}{1 - K \|x\|^2} x \in \mathbb{K}_K^n. \quad (3.66)$$

In particular, $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}(\mathbf{0}) = \mathbf{0}$. Given x in the hyperbolic model $\mathcal{H} \in \{\mathbb{K}_K^n, \mathbb{P}_K^n\}$ and tangent vector $v \in T_x \mathcal{H}$, the differential maps of $\pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n}$ and $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}$ are

$$(\pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n})_{*,x}(v) = \frac{1}{1 + \sqrt{1 + K \|x\|^2}} v - \frac{K \langle x, v \rangle}{\left(1 + \sqrt{1 + K \|x\|^2}\right)^2 \sqrt{1 + K \|x\|^2}} x,$$

3.3. Gyrogroup Batch Normalization

$$(\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n})_{*,x}(v) = \frac{2}{(1 - K \|x\|^2)}v + \frac{4K \langle x, v \rangle}{(1 - K \|x\|^2)^2}x.$$

In particular, the differential maps at the zero vector are

$$(\pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n})_{*,\mathbf{0}}(v) = \frac{1}{2}v, \quad (3.67)$$

$$(\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n})_{*,\mathbf{0}}(v) = 2v. \quad (3.68)$$

Moreover, these isometries preserve gyroaddition and gyromultiplication:

$$\begin{aligned} \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}(x \oplus_M y) &= \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}(x) \oplus_E \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}(y), \quad \forall x, y \in \mathbb{P}_K^n, \\ \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}(t \odot_M x) &= t \odot_E \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}(x), \quad \forall t \in \mathbb{R}, \forall x \in \mathbb{P}_K^n, \end{aligned} \quad (3.69)$$

where $\oplus_M$ and $\odot_M$ are Möbius operations, while $\oplus_E$ and $\odot_E$ are the Einstein counterparts.

Theorem 98 (Einstein by Beltrami–Klein). [$\Downarrow$] *The Einstein gyro operations can be rewritten as Eqs. (2.77) and (2.78):*

$$x \oplus_E y = \text{Exp}_x(\text{PT}_{\mathbf{0} \rightarrow x}(\text{Log}_{\mathbf{0}}(y))), \quad \forall x, y \in \mathbb{K}_K^n, \quad (3.70)$$

$$t \odot_E x = \text{Exp}_{\mathbf{0}}(t \text{Log}_{\mathbf{0}}(x)), \quad \forall x \in \mathbb{K}_K^n, \forall t \in \mathbb{R}. \quad (3.71)$$

Thm. 98 demonstrates that the Einstein gyro-structure can be expressed by the Beltrami–Klein geometry. Conversely, the Beltrami–Klein geometry can also be formulated by the Einstein gyro-structure.

Theorem 99 (Beltrami–Klein by Einstein). [$\Downarrow$] *Given $x, y \in \mathbb{K}_K^n$ and $v \in T_x \mathbb{K}_K^n$, the distance, exponential, and logarithmic operators under the Beltrami–Klein geometry are*

$$\begin{aligned} d(x, y) &= \frac{2}{\sqrt{|K|}} \tanh^{-1} \left(\sqrt{|K|} \frac{\| -x \oplus_E y \|}{1 + \sqrt{1 + K \| -x \oplus_E y \|^2}} \right), \\ \text{Exp}_x(v) &= x \oplus_E \text{Exp}_{\mathbf{0}} \left(\frac{1}{\sqrt{1 + K \|x\|^2}}v - \frac{K \langle x, v \rangle}{\left(1 + \sqrt{1 + K \|x\|^2}\right) (1 + K \|x\|^2)}x \right), \end{aligned}$$

$$\text{Log}_x(y) = \frac{1}{\lambda_K^x} (\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n})_{*, \tilde{x}} (\text{Log}_{\mathbf{0}}(-x \oplus_E y)),$$

where $\tilde{x} = \pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n}(x)$. In particular, the exponential and logarithmic maps at the zero vector $\mathbf{0}$ are identical across the Beltrami–Klein and Poincaré ball models:

$$\text{Exp}_{\mathbf{0}}(v) = \tanh(\sqrt{|K|} \|v\|) \frac{v}{\sqrt{|K|} \|v\|}, \quad \forall v \in T_{\mathbf{0}}\mathcal{H}, \quad (3.72)$$

$$\text{Log}_{\mathbf{0}}(x) = \tanh^{-1}(\sqrt{|K|} \|x\|) \frac{x}{\sqrt{|K|} \|x\|}, \quad \forall x \in \mathcal{H}, \quad (3.73)$$

with $\mathcal{H} \in \{\mathbb{K}_K^n, \mathbb{P}_K^n\}$.

Remark 100. Since the Beltrami–Klein and Poincaré ball models share the same $\text{Exp}_{\mathbf{0}}$ and $\text{Log}_{\mathbf{0}}$, it naturally follows that the Einstein and Möbius gyromultiplication coincide.

As the Beltrami–Klein model is isometric to the Poincaré ball, Thm. 78 implies gyroisometries.

Theorem 101. *On the Beltrami–Klein model, the gyrodistance is identical to the geodesic distance, whereas the gyroinverse, gyration, and left gyrotranslation are gyroisometries.*

Beltrami–Klein GyroBN. Thm. 101 ensures that GyroBN on the Beltrami–Klein model normalizes the sample mean and variance. Owing to the isometry between the Beltrami–Klein and Poincaré models, the Fréchet mean can be computed via the Poincaré ball: map the data to the Poincaré model using $\pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n}$, compute the Poincaré Fréchet mean [142, Alg. 1], and map the result back using $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}$. Together with the gyro and Riemannian operators in Sec. 3.3.4.6, we have all the ingredients to implement Alg. 2.

3.3.4.7 Correlation Manifolds

As reviewed in Sec. 2.9.2, ECM, LECM, OLM, and LSM are correlation metrics induced from Euclidean, zero-curvature prototype spaces, and these four metrics have been instantiated for LieBN in Sec. 3.2.5.3. Here, we further handle the nonzero-curvature correlation metric, namely PHCM. Under PHCM, any correlation matrix can be identified with a product of hyperbolic spaces via its Cholesky decomposition. Given $C \in \text{Cor}^+(n)$, let $L = \text{Chol}(C)$ be its Cholesky factor. The k -th row of L has

3.3. Gyrogroup Batch Normalization

the form $(L_{k1}, \dots, L_{k,k-1}, L_{kk}, 0, \dots, 0)$ with $L_{kk} > 0$, which belongs to the hyperbolic open hemisphere⁹

$$\text{HS}^{k-1} = \{x \in \mathbb{R}^k \mid \|x\| = 1, x_k > 0\}. \quad (3.74)$$

As detailed in Sec. 5.4, HS^n is isometric to the unit Poincaré ball

$$\mathbb{P}^n = \{x \in \mathbb{R}^n \mid \|x\| < 1\} \quad (3.75)$$

by

$$\pi_{\text{HS}^n \rightarrow \mathbb{P}^n} \left(\begin{bmatrix} x \\ x_{n+1} \end{bmatrix} \right) = \frac{x}{1 + x_{n+1}}. \quad (3.76)$$

Therefore, each correlation matrix can be identified with $n - 1$ Poincaré vectors:

$$\text{Cor}^+(n) \ni C \mapsto \begin{bmatrix} 1 & 0 & \cdots & 0 \\ L_{21} & L_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n1} & L_{n2} & \cdots & L_{nn} \end{bmatrix} \mapsto \begin{bmatrix} x_1 \in \mathbb{P}^1 \\ \vdots \\ x_{n-1} \in \mathbb{P}^{n-1} \end{bmatrix}. \quad (3.77)$$

Here, $x_i = \pi_{\text{HS}^i \rightarrow \mathbb{P}^i} (L_{(i+1,1)}, \dots, L_{(i+1,i+1)})^\top$ corresponds to the $(i + 1)$ -th row of the Cholesky factor. Let $\mathbb{PP}^{n-1} = \prod_{i=1}^{n-1} \mathbb{P}^i$ denote the product of unit Poincaré balls. We denote the identification in Eq. (3.77) by $\Phi : \text{Cor}^+(n) \rightarrow \mathbb{PP}^{n-1}$. GyroBN on the correlation manifold can then be realized via the Poincaré GyroBN applied row-wise: first map C to $\mathbb{PP}^{n-1}$ via Φ , apply GyroBN_i independently on each $\mathbb{P}^i$, and finally map back with Φ^{-1} . For a batch of activations $\{C^i\}_{i=1}^N \subset \text{Cor}^+(n)$, the process can be expressed as

$$\forall i \leq N, \quad C^i \xrightarrow{\Phi} \begin{bmatrix} x_1^i \in \mathbb{P}^1 \\ \vdots \\ x_{n-1}^i \in \mathbb{P}^{n-1} \end{bmatrix} \xrightarrow[\text{GyroBN}_{n-1}]{\text{GyroBN}_1} \begin{bmatrix} \tilde{x}_1^i \in \mathbb{P}^1 \\ \vdots \\ \tilde{x}_{n-1}^i \in \mathbb{P}^{n-1} \end{bmatrix} \xrightarrow{\Phi^{-1}} \tilde{C}^i. \quad (3.78)$$

3.3.4.8 Summary

To conclude this section, Tab. 3.14 summarizes the key gyro operators needed to implement GyroBN on representative manifolds. The correlation manifold is excluded, since its GyroBN is realized row-wise through the Poincaré ball.

⁹Also known as the Jemisphere model, where the “J” is pronounced as in Spanish [33, Sec. 7].

$U \oplus^{\text{Gr}} V$	$\ominus^{\text{Gr}} U$	$t \odot^{\text{Gr}} U$	Fréchet mean
$\exp(\Omega_U) V$	$\exp(-\Omega_U) I_{p,n}$	$\exp(t\Omega_U) I_{p,n}$	Karcher flow

(a) The ONB Grassmannian

Operator	$\mathfrak{st}_K^n$	$\mathcal{M}_K^n$	$\mathbb{K}_K^n$
$x \oplus y$	$\frac{(1 - 2K\langle x, y \rangle - K\ y\ ^2)x + (1 + K\ x\ ^2)y}{1 - 2K\langle x, y \rangle + K^2\ x\ ^2\ y\ ^2}$	Eq. (3.59)	Tab. 2.13
$\ominus x$	$-x$	$\begin{bmatrix} x_t \\ -x_s \end{bmatrix}$	$-x$
$t \odot x$	$\frac{\tan_K \left(t \tan_K^{-1} \left(\sqrt{ K } \ x\ \right) \right)}{\sqrt{ K }} \frac{x}{\ x\ }$	Eq. (3.55)	Tab. 2.13
Fréchet mean	$K < 0$: [142, Alg. 1] $K > 0$: Karcher flow	$K < 0$: [142, Alg. 3] $K > 0$: Karcher flow	Via Poincaré ball (see Sec. 3.3.4.6)

(b) Constant curvature spaces

Table 3.14: Summary of operators for GyroBN across representative manifolds.

3.3.5 Experiments

GyroBN layers are backbone-agnostic and can be integrated into different networks whenever the underlying manifold admits the required gyro operations. This subsection evaluates GyroBN on the Grassmannian, five CCSs, and the correlation manifold. The main findings are summarized as follows. More details on data sets and experimental settings are provided in Secs. A.1 and A.3.2.

- **Numerical experiments (Sec. 3.3.5.1).** The closed-form expression derived for radius gyroaddition in Eq. (3.59) significantly accelerates the computation compared with its definition-based counterpart in Eq. (3.53), achieving $2\times$ – $3\times$ speedups. Furthermore, visualizations demonstrate that GyroBN effectively normalizes sample distributions across diverse geometries.
- **Performance (Secs. 3.3.5.2 to 3.3.5.4).** On networks over the Grassmannian, CCSs, and the correlation manifold, GyroBN generally improves backbone networks, whereas existing RBN methods often degrade performance. Compared to these methods, GyroBN is generally faster or comparable in runtime, requires fewer or equal parameters, and improves robustness.

3.3.5.1 Numerical Experiments

Efficiency of the closed-form radius gyroaddition. Recalling Sec. 3.3.4.5, we derive closed-form expressions for the radius gyrooperations to improve computational

3.3. Gyrogroup Batch Normalization

Geometry	Hyperboloid		Sphere	
	Dim	Riemannian Closed form	Riemannian Closed form	
16	361.22	121.68 (33.69%)	323.58	122.02 (37.71%)
32	363.41	123.08 (33.87%)	327.75	123.71 (37.74%)
64	387.94	181.50 (46.78%)	453.42	181.05 (39.93%)
128	574.37	272.10 (47.37%)	644.18	270.21 (41.95%)
256	1149.58	534.09 (46.46%)	1137.62	538.58 (47.34%)
1024	3364.23	1414.67 (42.05%)	3551.69	1421.67 (40.03%)
2048	6479.95	2497.47 (38.54%)	6930.69	2449.56 (35.34%)

Table 3.15: Efficiency (in μs) of gyroaddition on the radius manifold: closed form versus Riemannian definition. Values in parentheses indicate the runtime of the closed-form implementation as a percentage of the corresponding Riemannian implementation. The best results are **bold**.

efficiency. To assess this, we compare two variants of radius gyroaddition: (i) the definition-based operator Eq. (3.53), implemented via a composition of the Riemannian logarithm, parallel transport, and exponential map; and (ii) the closed-form operator Eq. (3.59). We report the mean wall-clock time (in μs), averaged over 100 runs with a batch size of 10,000, across varying dimensions. As shown in Tab. 3.15, the closed-form implementation consistently outperforms its definition-based counterpart, achieving speedups of roughly $2\times$ – $3\times$ across the hyperboloid and sphere.

Visualization of GyroBN on different geometries. To intuitively illustrate the effect of GyroBN, we visualize its behavior on the ONB Grassmannian $\text{Gr}(1, 3)$, five CCSs with $|K| = 1$, and the correlation manifold $\text{Cor}^+(3)$. For each geometry, we randomly generate a batch of points, compute their batch mean, apply GyroBN, and then plot the normalized batch along with the resulting mean. The visualizations are constructed using the following embeddings:

- **Grassmannian.** Since $\text{Gr}(1, 3)$ is homeomorphic to the real projective space $\mathbb{RP}^2$, it is depicted as the unit hemisphere with antipodal points identified.
- **CCSs.** The unit Poincaré ball $\mathbb{P}^3$ and unit Beltrami–Klein ball $\mathbb{K}^3$ are shown as the interior of the unit ball in $\mathbb{R}^3$. The unit hyperboloid $\mathbb{H}^2$ is visualized as the upper sheet of a two-sheeted hyperboloid in $\mathbb{R}^3$. The unit sphere $\mathbb{S}^2$ is embedded as a 2-sphere in $\mathbb{R}^3$, while the projected hypersphere $\mathbb{D}^3$ coincides with $\mathbb{R}^3$ itself.
- **Correlation manifold.** $\text{Cor}^+(3)$ is embedded in $\mathbb{R}^3$ as an open ellipsope using its strictly lower triangular part.

For better visualization, we fix the bias parameter to the gyro identity element and set the scaling parameter to 0.4 for the Grassmannian, correlation manifold, and sphere; 0.7 for the Poincaré ball, Beltrami–Klein ball, and hyperboloid; and 1 for the projected

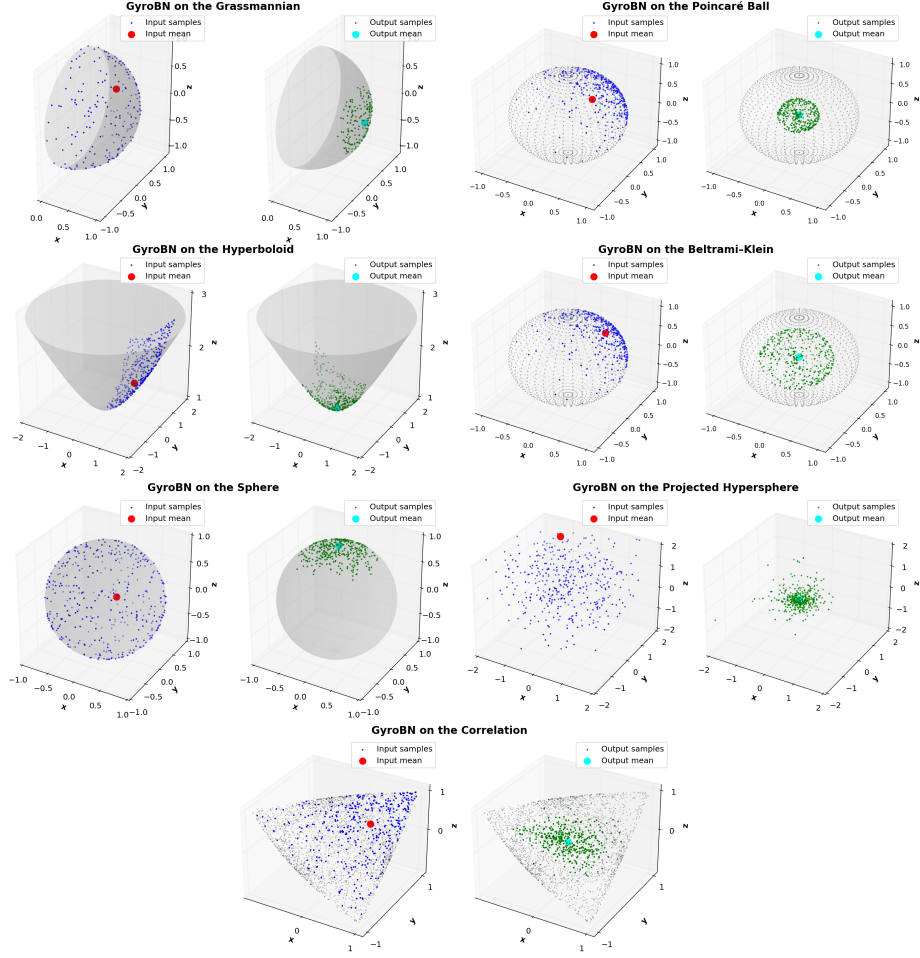


Figure 3.8: Visualization of GyroBN across different geometries. Blue and green points represent input and normalized data, respectively. Red and cyan points denote the input and output batch means. Black points mark the manifold boundary, and the gray surface depicts the manifold.

hypersphere. As shown in Fig. 3.8, GyroBN consistently normalizes data distributions across these geometries. Notably, although the Poincaré and Beltrami–Klein inputs are identical, their GyroBN behavior and resulting sample distributions differ due to their distinct Riemannian metrics, underscoring that GyroBN faithfully respects the underlying geometry.

3.3.5.2 Experiments on Grassmannian Neural Networks

Data sets and preprocessing. In line with previous work [108, 159], we evaluate GyroBN on three skeleton-based action recognition tasks, including HDM05 [153], NTU60 [178], and NTU120 [138] data sets, focusing on mutual actions for NTU60 and NTU120.

3.3. Gyrogroup Batch Normalization

Each sequence is represented as a Grassmannian matrix of size 93×10 , 150×10 , and 150×10 for HDM05, NTU60, and NTU120, respectively.

Comparative methods. Since no Grassmannian-specific BN methods exist, we adapt two previous approaches to the Grassmannian: ManifoldNorm [34, Algs. 1–2] and the RBN method of Lou et al. [142, Alg. 2], which we denote LRBN for clarity. Although neither was originally designed for the Grassmannian, they can be adapted by employing Riemannian operators such as geodesics, exponential/logarithmic maps, and parallel transport. *The key difference is that GyroBN can normalize data distributions across different geometries, whereas the other two methods cannot.*

Backbone networks. We adopt the recently proposed GyroGr architecture [159] as the backbone, which is briefly reviewed in Sec. A.2.2. GyroGr replaces the non-intrinsic FRMap + ReOrth block in GrNet [108] with Grassmannian left gyrotranslation, thereby improving numerical stability and performance. It consists of three basic components: left gyrotranslation, pooling [108], and the Projection Map (ProjMap) [108], where ProjMap maps Grassmannian points to symmetric matrices for classification. We consider both the 1-block and L -block variants. The 1-block version is structured as: gyrotranslation $\rightarrow$ pooling $\rightarrow$ ProjMap $\rightarrow$ classification, where the classification is implemented as an FC layer with softmax. The L -block version stacks L blocks of gyrotranslation and pooling, followed by a final ProjMap and classification layer. Since each pooling step approximately halves the dimensionality, we omit the pooling operation in the last block when $L > 1$. Following Huang et al. [108], the number of channels is fixed to 8.

Implementation details. Following Nguyen and Yang [159], we use the Cayley map to approximate the matrix exponential of skew-symmetric matrices and apply the trivialization strategy reviewed in Sec. 2.7 to parameterize the Grassmannian variables in both the gyrotranslation and GyroBN layers. Specifically, each trainable Grassmannian parameter is represented by Euclidean coordinates through the exponential map at the identity. This allows direct use of PyTorch optimizers [169] and avoids direct Riemannian updates of these parameters. In contrast, we find that the Grassmannian LRBN benefits from Riemannian optimization. Thus, we employ **Geoopt** [125] to optimize its Grassmannian bias parameter. Similarly, we use **Geoopt** to update the orthogonal bias parameter in ManifoldNorm. For all models, the BN layer is inserted after the first pooling layer with a momentum of 0.1. Training uses SGD with a learning rate of $5e^{-2}$, batch size 30, and 400, 200, and 200 epochs for HDM05, NTU60, and NTU120, respectively. All models are optimized with a standard cross-entropy loss. Following previous normalization methods on matrix manifolds [123, 215] and the LieBN implementation in Sec. A.3.1, we adopt a single Fréchet mean iteration.

Method	HDM05 (47 × 10)			NTU60 (75 × 10)			NTU120 (75 × 10)		
	Acc	Fit Time	#Params (M)	Acc	Fit Time	#Params (M)	Acc	Fit Time	#Params (M)
GyroGr	48.97±0.24	2.09	2.0744	70.13±0.16	28.16	0.5062	53.76±0.18	49.62	1.1812
GyroGr-ManifoldNorm	49.67±0.76	32.90	2.0921	68.56±0.43	232.60	0.5512	51.41±0.38	399.78	1.2262
GyroGr-LRBN	48.64±0.77	33.31	2.0781	67.77±0.52	238.53	0.5122	50.56±0.22	403.40	1.1872
GyroGr-GyroBN	51.89±0.37	3.04	2.0773	72.60±0.04	35.85	0.5114	55.47±0.10	67.37	1.1864

Table 3.16: Comparison of GyroBN against other Grassmannian BN methods under the GyroGr backbone. Here, accuracy is reported as a percentage, fit time denotes the average training time per epoch (s/epoch), and #Params is reported in millions. Values in parentheses specify the dimension of the Grassmannian input to the BN layer. The largest number of parameters is marked in red.

Main results. We compare GyroBN with ManifoldNorm and LRBN under the 1-block GyroGr backbone. The 5-fold results are presented in Tab. 3.16. We have the following four findings, which highlight the effectiveness of GyroBN in facilitating network training.

- **Improved accuracy.** Across all three data sets, GyroBN consistently improves performance, enhancing the accuracy of the vanilla GyroGr by 2.92, 2.47, and 1.71 percentage points on HDM05, NTU60, and NTU120, respectively. In contrast, both ManifoldNorm and LRBN often degrade performance, particularly on NTU60 and NTU120. This advantage comes from the theoretical guarantee of GyroBN for normalizing sample statistics, which is absent in the other two methods (see Tab. 3.13).
- **Enhanced efficiency.** GyroBN is substantially more efficient than ManifoldNorm and LRBN. The efficiency gain is mainly attributed to: (i) replacing computationally expensive Riemannian operators (e.g., parallel transport, exponential/logarithmic maps) with simpler gyro operations, (ii) reducing matrix multiplications from $n \times p$ to $(n - p) \times p$ or $p \times p$ (see Thm. 86), and (iii) applying trivialization to avoid costly Riemannian optimization.
- **Improved parameter economy.** GyroBN also requires fewer parameters than LRBN and ManifoldNorm. The key difference lies in the bias parameter. Due to the trivialization, GyroBN only needs an $(n - p) \times p$ Euclidean matrix, whereas LRBN requires an $n \times p$ Grassmannian matrix and ManifoldNorm requires an $n \times n$ orthogonal matrix.
- **Stronger generalization.** As shown in Fig. 3.9, we observe that GyroBN can narrow the gap between training and testing accuracy, indicating a stronger generalization ability.

3.3. Gyrogroup Batch Normalization

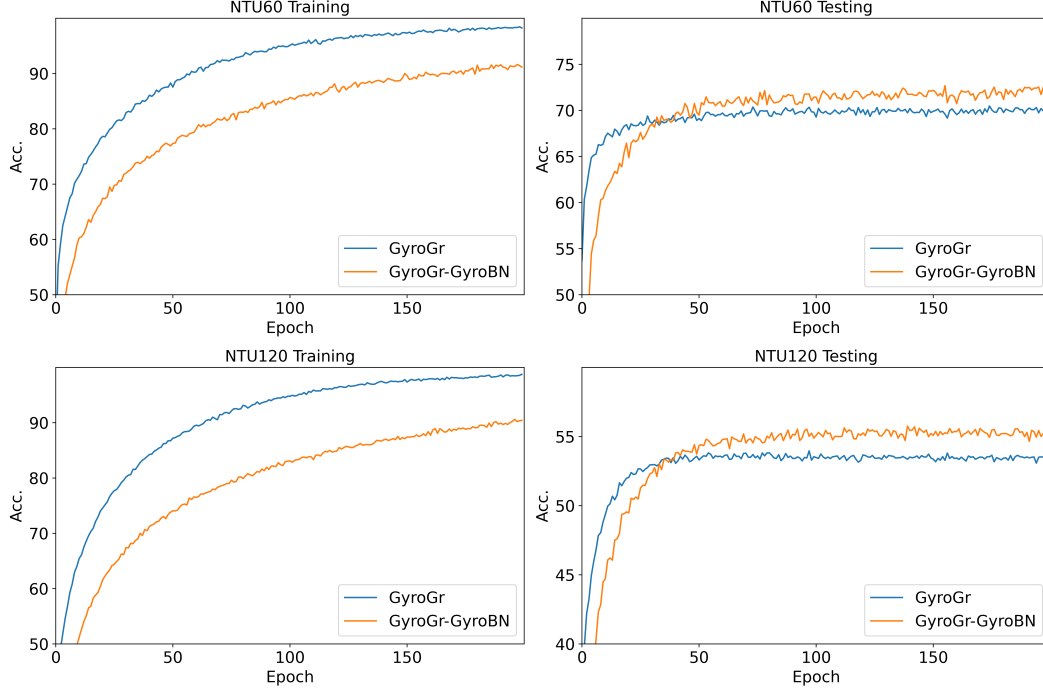


Figure 3.9: Training and testing curves of 1-block GyroGr on two NTU data sets.

3.3.5.3 Experiments on Networks over CCSs

Data sets. Following Lou et al. [142], we focus on the link prediction task on four graph data sets: Cora [177], Disease [4], Airport [229], and PubMed [155].

Comparative methods. We compare GyroBN with LRBN [142, Alg. 2]. As the original LRBN is only implemented in the Poincaré ball, we extend it to the other four CCSs. In particular, the core difference between LRBN and GyroBN lies in normalization: GyroBN can normalize sample statistics across different geometries, while LRBN lacks this guarantee.

Backbone networks. We use a *Hyperbolic Neural Network (HNN)* [76] for the Poincaré ball and Klein HNN (KNN) [147] for the Beltrami–Klein. For the sphere and projected hypersphere, we mimic the transformation and activation in HNN [76, Sec. 3.2] to build the corresponding layers. The layers on the four spaces above can be expressed as

$$\begin{aligned} \text{Transformation: } x^k &= \text{Exp}_e (M^k \text{Log}_e(x^{k-1})) \oplus b^k, \text{ with } b^k \in \mathcal{N}, \text{ and } M^k \in \mathbb{R}^{m \times n}, \\ \text{Activation: } x^k &= \text{Exp}_e (\phi (\text{Log}_e(x^{k-1}))), \text{ with } \phi \text{ as an activation,} \end{aligned}$$

where e is the origin, $\oplus$ is the gyroaddition, and $\mathcal{N} \in \{\mathbb{P}_K^n, \mathbb{K}_K^n, \mathbb{S}_K^n, \mathbb{D}_K^n\}$. For the

hyperboloid, we use the Lorentz fully-connected layer [45, Eq. 3] and Lorentz activation layer [15, Eq. 13], which are briefly reviewed in Sec. A.2.4. The above backbone network is referred to as HNN, KNN, SNN, PHNN, and LNN, respectively. We collectively call them *Constant Curvature Neural Networks (CCNNs)*.

Implementation details on CCNNs. We follow the official implementations of HGCN¹⁰ [38], LRBN¹¹ [142], and HCCN¹² [15] to conduct experiments, where we adopt the same training settings as Lou et al. [142, Sec. H.1]. Specifically, the baseline encoder is a CCNN with two transformation layers: the first maps the input feature dimension to 128, and the second maps 128 to 128. After each transformation layer, we use a ReLU activation, except for the Cora data set where activation is omitted. A BN layer, GyroBN or LRBN, is inserted after each transformation layer. The curvature is set as $|K| = 1$. For the manifold-valued bias parameter in GyroBN and LRBN, we apply the exponential map $\text{Exp}_e(v)$ to trivialize it via a Euclidean parameter v . The optimization is performed with Adam [121], using a learning rate of $1e^{-2}$ and a weight decay of $1e^{-3}$, except for the Cora data set, where weight decay is set to 0. The Fréchet mean iterations are performed until convergence.

Main results. We compare GyroBN with LRBN across five CCSs under the CCNN backbone. Tab. 3.17 reports the 5-fold average testing AUC on four data sets. We highlight the following findings.

- **Improved performance.** GyroBN consistently improves performance over the vanilla CCNNs across all data sets and geometries, whereas LRBN degrades performance in several cases (highlighted in red). The gains are especially pronounced on the sphere (SNN), where GyroBN achieves improvements of 17.65 (Disease), 7.49 (Airport), and 13.37 (PubMed) percentage points. This contrast underscores the advantage of GyroBN’s theoretical guarantee of normalizing sample statistics.
- **Efficiency.** As shown in the “Fit Time” rows of Tab. 3.17, GyroBN is more efficient than LRBN on the Poincaré ball, Beltrami–Klein and projected hypersphere, due to the simplicity of gyro operations. On the hyperboloid and sphere, GyroBN and LRBN exhibit comparable efficiency.
- **Parameter equivalence.** GyroBN and LRBN require the same number of parameters, only marginally more than those of the vanilla backbone. Thus, the

¹⁰<https://github.com/HazyResearch/hgcn>

¹¹<https://github.com/CUAI/Differentiable-Frechet-Mean>

¹²<https://github.com/kschwethelm/HyperbolicCV>

3.3. Gyrogroup Batch Normalization

Space		Poincaré Ball $\mathbb{P}_K^n$			Hyperboloid $\mathbb{H}_K^n$			Beltrami-Klein $\mathbb{K}_K^n$		
Method		HNN	HNN-LRBN	HNN-GyroBN	LNN	LNN-LRBN	LNN-GyroBN	KNN	KNN-LRBN	KNN-GyroBN
Disease	ROC	79.21 ± 2.14	76.58 ± 2.15	81.18 ± 0.93	87.71 ± 1.42	85.31 ± 0.95	88.87 ± 0.33	81.31 ± 1.37	78.98 ± 1.51	81.56 ± 0.70
	Fit Time	0.027	0.088	0.084	0.012	0.057	0.057	0.020	0.089	0.077
	#Params (M)	0.0180	0.0183	0.0183	0.0184	0.0187	0.0187	0.0180	0.0183	0.0183
Airport	ROC	94.63 ± 0.19	94.17 ± 0.40	95.40 ± 0.17	93.86 ± 0.21	93.05 ± 1.00	95.06 ± 0.12	95.00 ± 0.05	94.47 ± 0.33	96.14 ± 0.05
	Fit Time	0.054	0.122	0.119	0.047	0.089	0.092	0.056	0.136	0.126
	#Params (M)	0.0182	0.0184	0.0184	0.0186	0.0188	0.0188	0.0182	0.0184	0.0184
PubMed	ROC	95.02 ± 0.42	93.40 ± 0.20	95.83 ± 0.11	95.36 ± 0.10	95.88 ± 0.09	95.89 ± 0.11	95.87 ± 0.11	89.83 ± 0.27	96.23 ± 0.12
	Fit Time	0.125	0.342	0.335	0.111	0.242	0.249	0.125	0.353	0.337
	#Params (M)	0.0806	0.0809	0.0809	0.0815	0.0818	0.0818	0.0806	0.0809	0.0809
Cora	ROC	89.96 ± 0.44	93.47 ± 0.49	94.32 ± 0.22	91.84 ± 1.01	92.62 ± 0.17	93.66 ± 0.30	90.03 ± 0.32	93.38 ± 0.12	93.48 ± 0.25
	Fit Time	0.032	0.091	0.076	0.114	0.244	0.242	0.038	0.091	0.076
	#Params (M)	0.2001	0.2003	0.2003	0.2019	0.2021	0.2021	0.2001	0.2003	0.2003

(a) Results on three hyperbolic spaces.

Space		Projected Hypersphere $\mathbb{D}_K^n$			Sphere $\mathbb{S}_K^n$		
Method		PHNN	PHNN-LRBN	PHNN-GyroBN	SNN	SNN-LRBN	SNN-GyroBN
Disease	ROC	69.70 ± 2.01	60.25 ± 1.25	72.26 ± 0.61	54.19 ± 2.21	53.38 ± 4.07	71.84 ± 0.89
	Fit Time	0.022	0.066	0.063	0.029	0.042	0.043
	#Params (M)	0.0180	0.0183	0.0183	0.0181	0.0183	0.0183
Airport	ROC	89.60 ± 0.99	87.06 ± 0.46	90.44 ± 0.93	83.63 ± 0.77	86.14 ± 0.79	91.12 ± 1.57
	Fit Time	0.056	0.102	0.095	0.053	0.070	0.071
	#Params (M)	0.0182	0.0184	0.0184	0.0182	0.0184	0.0184
PubMed	ROC	89.86 ± 0.39	90.06 ± 0.23	92.06 ± 0.64	79.94 ± 1.76	90.10 ± 0.21	93.31 ± 0.08
	Fit Time	0.121	0.175	0.174	0.120	0.140	0.156
	#Params (M)	0.0806	0.0809	0.0809	0.0806	0.0809	0.0809
Cora	ROC	92.88 ± 0.26	92.03 ± 0.42	93.26 ± 0.42	92.10 ± 0.40	82.01 ± 0.71	93.16 ± 0.32
	Fit Time	0.026	0.067	0.063	0.025	0.044	0.045
	#Params (M)	0.2001	0.2003	0.2003	0.2001	0.2003	0.2003

(b) Results on two spherical spaces.

Table 3.17: Comparison of GyroBN against LRBN across five CCSs. ROC is the testing AUC reported as a percentage, fit time is measured in s/epoch, and #Params is reported in millions. When LRBN degenerates the backbone network, the results are highlighted with red.

performance gains of GyroBN cannot be attributed to parameter size, but rather to its principled normalization mechanism.

3.3.5.4 Experiments on Correlation Neural Networks

Data sets and preprocessing. Following the CorNet protocol in Sec. 5.4, we use the Radar, HDM05, and FPHA data sets and model each input sequence as multichannel correlation matrices.

Comparative methods. Similar to Sec. 3.3.5.3, we compare GyroBN against LRBN.

Backbone networks. We adopt the PHCM-based CorNet detailed in Sec. 5.4 as the backbone network. CorNet identifies each correlation matrix $C \in \text{Cor}^+(n)$ with a collection of Poincaré vectors and applies Poincaré layers. Specifically, C is mapped to the poly-Poincaré space $\mathbb{PP}^{n-1}$ via Eq. (3.77). The resulting multi-channel Poincaré vectors are then merged into a single Poincaré representation through β -concatenation [181, Sec. 3.3]. Then, a Poincaré FC layer followed by a Poincaré MLR layer constructs the network.

Implementation details. Since CorNet operates in the Poincaré geometry, both

Methods	Radar			HDM05			FPHA		
	Acc	Fit Time	#Params (M)	Acc	Fit Time	#Params (M)	Acc	Fit Time	#Params (M)
CorNet	96.56 $\pm$ 0.86	2.12	0.0438	82.26 $\pm$ 0.92	0.74	0.4091	90.03 $\pm$ 0.63	0.70	1.1210
CorNet-LRBN	92.85 $\pm$ 2.46	2.33	0.0439	N/A	1.64	0.4094	81.53 $\pm$ 0.72	1.12	1.1213
CorNet-GyroBN	97.67 $\pm$ 0.36	2.19	0.0439	80.82 $\pm$ 0.86	1.33	0.4094	92.88 $\pm$ 0.20	1.07	1.1213

Table 3.18: Comparison of CorNet with or without RBN layers. Accuracy is reported as a percentage, fit time is measured in s/epoch, and #Params is reported in millions.

GyroBN and LRBN are instantiated in the Poincaré model and applied after the Poincaré FC layer. To stabilize training, we scale the learning rate of the scaling parameter s by factors of 0.1 and 0.5 for Radar and HDM05, respectively. On FPHA, we further regularize the scaling by clamping: $\min\left(\frac{s}{\sqrt{v^2+\epsilon}}, 4\right)$. For better efficiency, the number of Fréchet mean iterations is set to 2.

Main results. We summarize the comparison of CorNet with or without normalization layers in Tab. 3.18. Overall, GyroBN demonstrates clear benefits on Radar and FPHA with negligible parameter cost and modest efficiency trade-offs. On HDM05, however, neither GyroBN nor LRBN improves the baseline, with LRBN even diverging and failing to converge.

3.4 Conclusion

This chapter developed a unified approach to RBN across broad families of manifolds. We first presented LieBN, which operates under left-, right-, and bi-invariant metrics. LieBN uses the group translations for centering and biasing and performs scaling in the tangent space at the identity element. These operations provide theoretical control over Riemannian sample and population statistics. Their instantiations on SPD, rotation, and full-rank correlation manifolds further demonstrate how a choice of Lie group structure and invariant metric yields a concrete normalization layer, including the power-deformed SPD structures and CRIM developed in this chapter.

The chapter then introduced pseudo-reductive gyrogroups as a relaxation of classical gyrogroups, providing a broader algebraic foundation for Riemannian normalization. Building on this structure, we developed GyroBN and showed that gyroisometric gyrations enable theoretical control over sample statistics. Since every Lie group is a gyrogroup with identity gyrations, GyroBN recovers LieBN as a special case. GyroBN replaces group subtraction, addition, and tangent-space scaling with gyrosubtraction, gyroaddition, and scalar gyromultiplication, thereby extending the normalization principle to broader geometries. Finally, we instantiated the framework on the Grassmannian, five CCS models, and the full-rank correlation manifold. Experiments across all

considered geometries support the effectiveness of this extension.

Together, LieBN and GyroBN extend normalization with theoretical control of intrinsic statistics from Lie groups to pseudo-reductive gyrogroups. Gyrogroups substantially broaden the scope beyond Lie groups but do not encompass all manifolds. A natural direction for future work is therefore to extend this normalization principle to manifolds that do not admit suitable gyro-structures. The next chapter considers intrinsic classification.

Chapter 4

Riemannian Multinomial Logistic Regression

4.1 Introduction

Although Riemannian networks demonstrated success in many applications, most approaches still rely on Euclidean spaces for classification, such as tangent spaces [106, 107, 31, 156, 207, 209, 157, 158, 123, 208, 47], ambient Euclidean spaces [205, 183, 184], or coordinate systems [36].¹ However, these strategies distort the intrinsic geometry of the manifold, undermining the effectiveness of Riemannian networks. Researchers have recently started directly developing *Riemannian Multinomial Logistic Regression (RMLR)* on manifolds. Inspired by the idea of hyperplane margin [130], Ganea et al. [76] developed a hyperbolic MLR in the Poincaré ball for HNNs. Motivated by HNNs, Nguyen and Yang [159] developed three kinds of gyro SPD MLRs based on three distinct gyro-structures of the SPD manifold. Nguyen et al. [161] proposed gyro MLRs for the *Symmetric Positive Semidefinite (SPSD) manifold* based on the product of gyro spaces. However, these classifiers often rely on manifold-specific geometric structures, limiting their generalizability to other geometries. For instance, the hyperbolic MLR [76] relies on the generalized law of sines, while the gyro MLRs [159, 161] rely on gyro-structures.

This chapter proceeds in two stages. We first study SPD manifolds endowed with pullback Euclidean metrics, whose flat geometry reduces the point-to-hyperplane infimum to a Euclidean problem and yields closed-form intrinsic MLRs. We then extend the classification principle to general Riemannian manifolds. Rather than evaluating

¹Notice that there are also works designing classifiers for grid-based manifold-valued data [37], but we focus on non-gridded data in line with many previous SPD networks.

the potentially intractable point-to-hyperplane infimum on each manifold, the general RMLR adopts a Riemannian-trigonometric formulation that requires only a well-defined Riemannian logarithm. Since the SPD classifiers in Sec. 4.2 are special cases of the general RMLR framework in Sec. 4.3, and the two parts use the same experimental settings, all experiments² are presented together in Sec. 4.4.

Throughout this chapter and its accompanying appendix material, all parameters are assumed to satisfy the standing admissibility conditions.³

4.2 Multinomial Logistic Regression on SPD Manifolds

4.2.1 Introduction

SPD matrices are commonly encountered in a diverse range of scientific fields, such as medical imaging [36, 37], signal processing [8, 105, 32, 31], elasticity [152, 89], question answering [141, 157], graph classification [42], and computer vision [106, 94, 232, 34, 41, 230, 37, 48, 183, 156, 158, 185]. Despite their ubiquitous presence, traditional learning algorithms are ineffective in handling the non-Euclidean geometry of SPD matrices. To address this limitation, several Riemannian metrics [171, 9, 137] have been proposed. With these Riemannian metrics, various machine learning techniques can be generalized to SPD manifolds.

Inspired by the great success of deep learning [102, 127, 97], several deep networks have been developed on SPD manifolds. Despite their promising performance, many approaches still rely on Euclidean spaces for classification, such as tangent spaces [106, 31, 156, 207, 157, 158, 123, 208, 47], ambient Euclidean spaces [205, 183, 184], and coordinate systems [36]. However, these strategies distort the intrinsic geometry of the SPD manifold, undermining the effectiveness of SPD neural networks. Notably, there are also some similarity-based classifiers originally designed for shallow learning methods [77, 94, 48]. Although these classifiers can be extended to deep SPD neural networks [209, 211], the calculation of pairwise distances might undermine training efficiency. Recently, motivated by HNNs [76], three kinds of SPD *Multinomial Logistic Regression* (MLR) classifiers based on the gyro-structures induced by LEM, LCM, and

²The code is available at <https://github.com/GitZH-Chen/RMLR>.

³All normal vectors are nonzero. Whenever a displayed formula contains $1/\theta$ or $1/\theta^2$, we assume $\theta \neq 0$, except when an explicit limit as $\theta \rightarrow 0$ is considered. The inner-product parameters satisfy $(\alpha, \beta) \in \mathbf{ST}$, as defined in Eq. (2.93). These conditions are not repeated below.

AIM were developed by Nguyen and Yang [159]. However, the proposed SPD MLRs rely on the gyro-structures, limiting their generality. Besides, Chakraborty et al. [37] also introduced an invariant layer for manifold-valued data mimicking the invariant FC layer in *Convolutional Neural Networks (CNNs)*. However, it is designed for gridded manifold-valued data, which is not the primary data type encountered in many other SPD networks. Following the convention of most SPD networks, we only focus on non-gridded cases.

In fact, SPD MLR can be directly derived under LEM and LCM without the assistance of gyro-structures. More generally, LEM and LCM are pullback Euclidean metrics, which are metrics pulled back from the Euclidean space. This section develops a unified construction of SPD MLR across pullback Euclidean metrics. On the empirical side, we focus on the parameterized LEM and LCM defined in Sec. 3.2.5.1, which generalize the standard LEM and LCM by the pullback of matrix power. Their deformation behavior is established in Thm. 67. We showcase our SPD MLRs under these parameterized metrics. Besides, our framework encompasses the gyro SPD MLRs induced by the standard LEM and LCM in Nguyen and Yang [159]. More importantly, our framework also provides an intrinsic explanation for the commonly used LogEig classifier on SPD manifolds, which consists of successive matrix logarithm, FC, and softmax layers. The main **contributions** are summarized as follows:

- (1) We develop a unified construction of SPD MLR under pullback Euclidean metrics and instantiate it under two parameterized metric families.
- (2) Our framework offers an intrinsic explanation of the most popular LogEig classifier, which stacks matrix logarithm, the FC layer, and softmax.

Outline. Sec. 4.2.2 develops SPD MLRs under pullback Euclidean metrics. The deformed-metric instantiation follows in Sec. 4.2.3. Sec. 4.2.4 reinterprets the existing LogEig classifier intrinsically. The corresponding experiments are presented together with the general RMLR experiments in Sec. 4.4. Proofs are deferred to Sec. B.4.

4.2.2 SPD Multinomial Logistic Regression

This section first reformulates the Euclidean MLR. Then, we deal with SPD MLR under an arbitrary pullback Euclidean metric on SPD manifolds. We use the pullback metric in Thm. 33 with a Euclidean codomain. The induced abelian Lie-group structure, bi-invariant metric, distance, and Riemannian operators required below are established later in Thm. 149.

4.2.2.1 Reformulation of Euclidean MLR

The Euclidean MLR was first reformulated by Lebanon and Lafferty [130] from the perspective of distances to margin hyperplanes. Hyperbolic MLR was designed based on this reformulation [76]. Nguyen and Yang [159] further proposed three gyro SPD MLRs based on the gyro-structures induced by AIM, LEM, and LCM. We now briefly review the reformulation of Euclidean MLR.

Given C classes, MLR in $\mathbb{R}^n$ computes the following softmax probabilities:

$$\forall k \in \{1, \dots, C\}, \quad p(y = k \mid x) \propto \exp(\langle a_k, x \rangle - b_k), \quad (4.1)$$

where $b_k \in \mathbb{R}$ and $x, a_k \in \mathbb{R}^n$. As shown in Lebanon and Lafferty [130, Sec. 5] and Ganea et al. [76, Sec. 3.1], Eq. (4.1) can be reformulated as

$$p(y = k \mid x) \propto \exp(\text{sign}(\langle a_k, x - p_k \rangle) \|a_k\| d(x, H_{a_k, p_k})), \quad (4.2)$$

where $\langle a_k, p_k \rangle = b_k$, and H_{a_k, p_k} is referred to as a hyperplane, defined as

$$H_{a_k, p_k} = \{x \in \mathbb{R}^n \mid \langle a_k, x - p_k \rangle = 0\}. \quad (4.3)$$

As reviewed in Tab. 2.3, $\text{Log}_p x$ is the natural generalization of the directional vector $\vec{px} = x - p$ starting at p and ending at x , while the Riemannian metric at p corresponds to the inner product. Therefore, the MLR in Eq. (4.2) and hyperplane in Eq. (4.3) can be readily generalized to the SPD manifold $(\mathcal{S}_{++}^n, g)$.

Definition 102 (SPD hyperplanes). Given $P \in \mathcal{S}_{++}^n$ and $A \in T_P \mathcal{S}_{++}^n \setminus \{\mathbf{0}\}$, we define the *SPD hyperplane* as

$$\tilde{H}_{A, P} = \{S \in \mathcal{S}_{++}^n \mid g_P(\text{Log}_P S, A) = \langle \text{Log}_P S, A \rangle_P = 0\}, \quad (4.4)$$

where P and A are referred to as shift and normal matrices, respectively.

Definition 103 (SPD MLR). The *SPD MLR* is defined as

$$p(y = k \mid S) \propto \exp\left(\text{sign}\left(\langle A_k, \text{Log}_{P_k}(S) \rangle_{P_k}\right) \|A_k\|_{P_k} d\left(S, \tilde{H}_{A_k, P_k}\right)\right), \quad (4.5)$$

where $P_k \in \mathcal{S}_{++}^n$, $A_k \in T_{P_k} \mathcal{S}_{++}^n \setminus \{\mathbf{0}\}$, $\langle \cdot, \cdot \rangle_{P_k} = g_{P_k}$, $\|\cdot\|_{P_k}$ is the norm on $T_{P_k} \mathcal{S}_{++}^n$ induced by g at P_k , and $\tilde{H}_{A_k, P_k}$ is a margin hyperplane in $\mathcal{S}_{++}^n$ as defined in Eq. (4.4).

$d(S, \tilde{H}_{A_k, P_k})$ denotes the margin distance between S and the SPD hyperplane $\tilde{H}_{A_k, P_k}$, which is formulated as

$$d(S, \tilde{H}_{A_k, P_k}) = \inf_{Q \in \tilde{H}_{A_k, P_k}} d(S, Q), \quad (4.6)$$

where $d(S, Q)$ is the geodesic distance induced by g .

In geometry, the hyperplane in Eq. (4.3) is actually a regular submanifold of the trivial manifold $\mathbb{R}^n$. As for our definition of SPD hyperplanes, we have a similar result.

Proposition 104 (Submanifolds). $\downarrow$ *The SPD hyperplane, as defined in Eq. (4.4), is a regular submanifold of the SPD manifold if $\text{Log}_P : \mathcal{S}_{++}^n \rightarrow T_P \mathcal{S}_{++}^n$ is a global diffeomorphism.*

Thm. 104 rationalizes Thm. 102, as both the SPD hyperplane and Euclidean hyperplane are submanifolds. Nevertheless, we still follow the nomenclature of Ganea et al. [76], Lebanon and Lafferty [130] and call $\tilde{H}_{A, P}$ an SPD hyperplane.

4.2.2.2 SPD MLRs under Pullback Euclidean Metrics

For our SPD MLR in Thm. 103, under most Riemannian metrics on SPD manifolds, all the operators involved in Eq. (4.5) have closed-form expressions except the margin distance in Eq. (4.6). Therefore, the only difficulty lies in the calculation of the margin distance. This subsection proposes a general expression for SPD MLRs under pullback Euclidean metrics, which are defined by a pullback from Euclidean metric, as detailed in Thm. 149. We choose the identity matrix I as a fixed anchor.

We choose pullback Euclidean metrics as our starting metrics mainly because of their extensive inclusion and easy computation. Several Riemannian metrics, including LEM, LCM, and their variants [196, 194], are pullback Euclidean metrics. Besides, due to their fast and simple calculation, the margin distance under a pullback Euclidean metric has a closed-form expression, while obtaining distances to hyperplanes under other metrics, such as AIM, would be complicated.

We start by calculating the margin distance in Eq. (4.6) under a given pullback Euclidean metric.

Lemma 105. $\downarrow$ *Given a pullback Euclidean metric g , the margin distance defined*

in Eq. (4.6) has a closed-form solution:

$$d(S, \tilde{H}_{A_k, P_k}) = d(\phi(S), H_{\phi_{*, P_k}(A_k), \phi(P_k)}) \quad (4.7)$$

$$= \frac{|\langle \phi(S) - \phi(P_k), \phi_{*, P_k}(A_k) \rangle|}{\|A_k\|_{P_k}}, \quad (4.8)$$

where $|\cdot|$ is the absolute value.

Putting Eq. (4.8) into Eq. (4.5), we obtain our SPD MLR under a given pullback Euclidean metric:

$$p(y = k | S) \propto \exp(\langle A_k, \text{Log}_{P_k}(S) \rangle_{P_k}) \quad (4.9)$$

$$= \exp(\langle \phi(S) - \phi(P_k), \phi_{*, P_k}(A_k) \rangle), \quad (4.10)$$

where $S, P_k \in \mathcal{S}_{++}^n$ and $A_k \in T_{P_k}\mathcal{S}_{++}^n \setminus \{0\}$. When P_k is fixed, $A_k \in T_{P_k}\mathcal{S}_{++}^n$ indeed lies in a Euclidean space. However, P_k would vary during training, making A_k non-Euclidean. To remedy this issue, A_k can be generated from a Euclidean parameter in a fixed tangent space through one of three mechanisms: Riemannian parallel transport [69], vector transport⁴ [1], or the differential of a group translation, including Lie group translation [197, Sec. 20] and gyrogroup translation [199]. We focus on Riemannian parallel transport and the differential of a Lie group translation, for which we establish an equivalence under pullback Euclidean metrics. Under parallel transport, we write $A_k = \text{PT}_{Q \rightarrow P_k}(\tilde{A}_k)$ with $\tilde{A}_k \in T_Q\mathcal{S}_{++}^n$ as a Euclidean parameter. This is the solution also adopted by HNNs [76], where the tangent point is the zero vector. Since the Lie groups associated with pullback Euclidean metrics are abelian, we only consider the left translation. We have the following two lemmas to show the relation between parallel transport and the differential of left translation.

Lemma 106. [↓] *Given a pullback Euclidean metric, any parallel transport is equivalent to the differential map of a left translation and vice versa.*

Lemma 107. [↓] *Given two fixed SPD matrices $Q_1, Q_2 \in \mathcal{S}_{++}^n$, we have the follow-*

⁴Vector transport is often used in optimization as a substitute for parallel transport because its expressions are typically simpler and cheaper to compute [27, Sec. 10.5].

ing equivalence for parallel transports under a pullback Euclidean metric:

$$\begin{aligned} \forall \tilde{A}_{1,k} \in T_{Q_1} \mathcal{S}_{++}^n, \exists! \tilde{A}_{2,k} \in T_{Q_2} \mathcal{S}_{++}^n, \\ \text{s.t. } \text{PT}_{Q_1 \rightarrow P_k}(\tilde{A}_{1,k}) = \text{PT}_{Q_2 \rightarrow P_k}(\tilde{A}_{2,k}). \end{aligned} \quad (4.11)$$

Thm. 106 indicates that under pullback Euclidean metrics, the above two solutions are equivalent, while Thm. 107 implies that anchor points can be arbitrarily chosen. Therefore, without loss of generality, we generate A_k from the tangent space at the identity matrix I by parallel transport, i.e., $A_k = \text{PT}_{I \rightarrow P_k}(\tilde{A}_k)$ with $\tilde{A}_k \in T_I \mathcal{S}_{++}^n \cong \mathcal{S}^n$. Together with Eq. (6.9), Eq. (4.10) can be further simplified.

Theorem 108 (SPD MLR under a pullback Euclidean metric). $\downarrow$ Under any pullback Euclidean metric, SPD MLR and SPD hyperplane are

$$p(y = k | S) \propto \exp \left(\left\langle \phi(S) - \phi(P_k), \phi_{*,I}(\tilde{A}_k) \right\rangle \right), \quad (4.12)$$

$$\tilde{H}_{\tilde{A}_k, P_k} = \left\{ S \in \mathcal{S}_{++}^n \mid \left\langle \phi(S) - \phi(P_k), \phi_{*,I}(\tilde{A}_k) \right\rangle = 0 \right\}, \quad (4.13)$$

where $\tilde{A}_k \in T_I \mathcal{S}_{++}^n \setminus \{0\} \cong \mathcal{S}^n \setminus \{0\}$ is a symmetric matrix, and $P_k \in \mathcal{S}_{++}^n$ is an SPD matrix.

4.2.3 SPD MLRs under Deformed LEM and LCM

In this section, we first review the deformed LEM and LCM, and then showcase our SPD MLR in Thm. 108 under these deformed metrics.

As reviewed in Sec. 3.2.5.1, inspired by the deforming utility of the matrix power function [192, 194], we define (θ, α, β) -LEM and θ -LCM as the pullback metrics of (α, β) -LEM and LCM by the matrix power function $(\cdot)^\theta$ and scaled by $\frac{1}{\theta^2}$ for $\theta \neq 0$. As shown in Thm. 67, (θ, α, β) -LEM is equal to (α, β) -LEM and θ -LCM interpolates between the standard LCM for $\theta = 1$ and an LEM-like metric as $\theta \rightarrow 0$.

Besides, both (α, β) -LEM and θ -LCM are pullback Euclidean metrics, as summarized in Tab. 3.6. Therefore, the SPD MLRs under these two families of metrics can be directly obtained by Thm. 108.

Corollary 109 (SPD MLRs under the deformed LEM and LCM). $\downarrow$ The SPD

MLR under (α, β) -LEM is

$$p(y = k \mid S) \propto \exp \left[\left\langle \log(S) - \log(P_k), \tilde{A}_k \right\rangle^{(\alpha, \beta)} \right], \quad (4.14)$$

where $\tilde{A}_k \in T_I \mathcal{S}_{++}^n \cong \mathcal{S}^n$ and $P_k \in \mathcal{S}_{++}^n$. The SPD MLR under θ -LCM is

$$p(y = k \mid S) \propto \exp \left[\frac{1}{\theta} \langle X, Y \rangle \right], \quad (4.15)$$

with X and Y defined as

$$X = \lfloor \tilde{K} \rfloor - \lfloor \tilde{L}_k \rfloor + \left[\text{Dlog} \left(\mathbb{D}(\tilde{K}) \right) - \text{Dlog} \left(\mathbb{D}(\tilde{L}_k) \right) \right], \quad (4.16)$$

$$Y = \lfloor \tilde{A}_k \rfloor + \frac{1}{2} \mathbb{D}(\tilde{A}_k), \quad (4.17)$$

where $\tilde{K} = \text{Chol}(S^\theta)$, $\tilde{L}_k = \text{Chol}(P_k^\theta)$, and $\mathbb{D}(\tilde{A}_k)$ denotes a diagonal matrix with the diagonal elements of $\tilde{A}_k$.

$\mathcal{S}_{++}^2$ can be visualized as an open cone in $\mathbb{R}^3$ by the condition that $P = \begin{pmatrix} x & y \\ y & z \end{pmatrix} \in \mathcal{S}^2$ is positive definite if and only if $x, z > 0 \wedge xz > y^2$. Fig. 4.1 illustrates SPD hyperplanes induced by (α, β) -LEM and θ -LCM.

Remark 110. This construction incorporates the results with respect to LEM and LCM presented by Nguyen and Yang [159]. For (α, β) -LEM, when $(\alpha, \beta) = (1, 0)$, (α, β) -LEM becomes the standard LEM. Our margin distance to the hyperplane in Thm. 105 becomes the pseudo-gyrodistance under LEM [159, Thm. 2.23]. For θ -LCM, when $\theta = 1$, θ -LCM becomes the standard LCM. Thm. 105 then becomes the pseudo-gyrodistance induced by LCM [159, Thm. 2.24]. However, our framework does not require gyro-structures and directly obtains the margin distance and SPD MLR based on the Riemannian metric.

4.2.4 Rethinking the Existing LogEig Classifier

Many existing SPD neural networks [106, 31, 160, 207, 156, 208, 47] rely on a Euclidean MLR in the codomain of matrix logarithm, *i.e.*, a matrix logarithm followed by an FC layer and a softmax layer. For simplicity, we call this classifier LogEig MLR. The existing explanation of LogEig MLR is that it approximates manifolds by a tangent

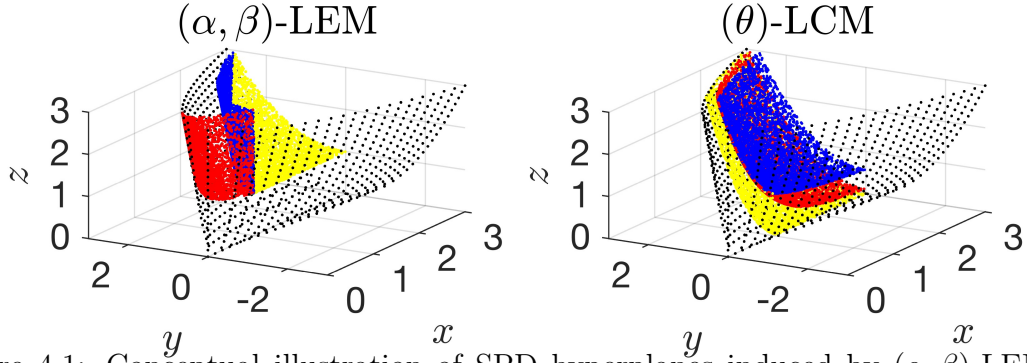


Figure 4.1: Conceptual illustration of SPD hyperplanes induced by (α, β) -LEM and θ -LCM. In each subfigure, the black dots are SPSP matrices, denoting the boundary of $\mathcal{S}^2_{++}$, while the blue, red, and yellow dots denote three SPD hyperplanes.

space. However, our framework can offer a novel intrinsic explanation for this widely used MLR.

When $(\alpha, \beta) = (1, 0)$ for (α, β) -LEM, the SPD MLR in Eq. (4.14) is very similar to the LogEig MLR. However, due to the nonlinearity of $\log(\cdot)$ and the non-Euclidean of the SPD parameter P_k , SPD MLR cannot be hastily viewed as equivalent to LogEig MLR. Nevertheless, under special circumstances, Eq. (4.14) is indeed equivalent to a LogEig MLR.

Proposition 111. $\Downarrow$ *The LEM-based SPD MLR is equivalent to a LogEig MLR with parameters in the FC layer optimized by Euclidean SGD when SPD manifolds are endowed with the standard LEM, the SPD parameter P_k in Eq. (4.14) is optimized by LEM-based RSGD, and the Euclidean parameter $\tilde{A}_k$ is optimized by Euclidean SGD.*

Thm. 111 implies that, when optimized by LEM-based RSGD, the LEM-based SPD MLR is equivalent to the Euclidean MLR in the codomain of matrix logarithm. Nevertheless, a substantial body of prior work underscores the theoretical and empirical superiority of AIM-based optimization over its LEM-based counterpart [186, 92]. Therefore, we adopt the AIM-based optimizer to update the involved SPD parameters.

4.3 Extension to General Riemannian Manifolds

4.3.1 Introduction

The SPD MLR framework developed in Sec. 4.2 constructs an intrinsic classifier from the geodesic distance between an SPD matrix and a margin hyperplane. Its central

quantity is the point-to-hyperplane distance in Eq. (4.6), defined as the infimum of the geodesic distance over all points on the hyperplane. For the pullback Euclidean metrics considered in Sec. 4.2.2.2, this optimization admits a closed-form solution. On a general Riemannian manifold, however, directly evaluating this infimum can require solving a difficult, potentially non-convex optimization problem, which prevents an extension of the preceding framework.

We circumvent this obstacle by reinterpreting the Euclidean margin distance from a trigonometric perspective rather than directly solving the infimum. This alternative characterization expresses the margin distance through the angle between geodesics and the geodesic distance from the input to the hyperplane anchor. Lifting this characterization to Riemannian manifolds yields a closed-form RMLR. Accordingly, the resulting framework only requires an explicit expression of the Riemannian logarithm, which is the minimal geometric requirement for extending Euclidean MLR to manifolds. Since this requirement is satisfied by many manifolds commonly encountered in machine learning, the framework applies broadly across different geometries. This reliance on an operator shared by many manifolds, rather than on additional manifold-specific structure, makes RMLR a unified classification module across geometries.

We instantiate the framework on SPD manifolds and rotation matrices. On the SPD manifold, we systematically develop SPD MLRs under five families of power-deformed metrics and provide a complete theoretical discussion of their geometric properties. On the Lie group $SO(n)$, we construct a Lie MLR under the widely used bi-invariant metric, providing the first extension of Euclidean MLR to Lie groups. Moreover, the framework incorporates several existing Riemannian MLRs, including gyro SPD MLRs [159], the SPD MLRs developed in Sec. 4.2, and gyro SPSP MLRs [161].

Our SPD MLRs are validated on four SPD backbone networks, including SPDNet [106] on the radar and human action recognition tasks and TSMNet [123] on the EEG classification tasks for the Riemannian feedforward network, RResNet [117] on the human action recognition task for the Riemannian residual network, and SPDGCN [231] on the node-classification task for the Riemannian graph neural network. Our Lie MLR is validated on the classic LieNet [107] backbone for the human action recognition task. Compared with previous non-intrinsic classifiers, our MLRs achieve consistent performance gains. In particular, our SPD MLRs outperform the previous classifiers by **14.23 percentage points** on SPDNet and **13.72 percentage points** on RResNet for human action recognition, and **4.46 percentage points** on TSMNet for EEG inter-subject classification. Furthermore, our Lie MLR can improve both the training stability and performance. In summary, our **main theoretical contributions** are the

following:

- (1) We develop a unified RMLR framework for general Riemannian manifolds that requires only the Riemannian logarithm and incorporates several existing manifold-specific RMLRs as special cases.
- (2) We systematically propose five families of SPD MLRs based on different geometries of the SPD manifold.
- (3) We propose a novel Lie MLR for deep neural networks on $\text{SO}(n)$.

Outline. Sec. 4.3.2 revisits the existing Riemannian MLRs and proposes the general RMLR framework. Sec. 4.3.3 instantiates the framework on SPD manifolds under five families of power-deformed metrics. Sec. 4.3.4 presents the Lie MLR on $\text{SO}(n)$. Sec. 4.4 reports experiments on Riemannian feedforward, residual, and graph networks, direct SPD classification, and LieNet. Proofs are deferred to Sec. B.5.

4.3.2 Riemannian Multinomial Logistic Regression

Inspired by Lebanon and Lafferty [130], prior work extended the Euclidean MLR to hyperbolic, SPD, and SPSD manifolds [76, 159, 161]. Sec. 4.2 further develops SPD MLRs under pullback Euclidean metrics. However, these classifiers rely on specific Riemannian properties, such as the generalized law of sines, gyro-structures, and flat metrics, which limit their generality. In this section, we first revisit several existing MLRs and then propose our Riemannian classifiers with minimal geometric requirements.

4.3.2.1 Revisiting Existing Multinomial Logistic Regressions

The Euclidean MLR and its reformulation through margin distances to hyperplanes have already been reviewed in Sec. 4.2.2.1. By abstracting the expressions for the SPD MLR and SPD hyperplanes in Eqs. (4.4) and (4.5), we can readily extend this construction from the SPD manifold to a general Riemannian manifold $\mathcal{M}$:

$$p(y = k \mid S) \propto \exp \left(\text{sign}(\langle \tilde{A}_k, \text{Log}_{P_k}(S) \rangle_{P_k}) \|\tilde{A}_k\|_{P_k} \tilde{d}(S, \tilde{H}_{\tilde{A}_k, P_k}) \right), \quad (4.18)$$

$$\tilde{H}_{\tilde{A}_k, P_k} = \left\{ S \in \mathcal{M} \mid g_{P_k}(\text{Log}_{P_k} S, \tilde{A}_k) = 0 \right\}, \quad (4.19)$$

where $P_k \in \mathcal{M}$, $\tilde{A}_k \in T_{P_k} \mathcal{M} \setminus \{\mathbf{0}\}$, g_{P_k} is the Riemannian metric at P_k , and Log_{P_k} is the Riemannian logarithm at P_k . The margin distance is defined as an infimum:

$$\tilde{d}(S, \tilde{H}_{\tilde{A}_k, P_k}) = \inf_{Q \in \tilde{H}_{\tilde{A}_k, P_k}} d(S, Q). \quad (4.20)$$

The MLRs in Lebanon and Lafferty [130], Ganea et al. [76], Nguyen and Yang [159] and Sec. 4.2 can be viewed as different implementations of Eqs. (4.18) to (4.20). To calculate the MLR in Eq. (4.18), one has to compute the associated Riemannian metrics, logarithmic maps, and margin distance. The associated Riemannian metrics and logarithmic maps often have closed-form expressions on the frequently encountered manifolds in machine learning. However, the computation of the margin distance can be challenging. On the Poincaré ball of hyperbolic manifolds, the generalized law of sines simplifies the calculation of Eq. (4.20) [76]. However, the generalized law of sines is not universally guaranteed on other manifolds. For pullback Euclidean metrics on SPD manifolds, Thm. 105 provides a closed-form solution for the margin distance. For curved manifolds, solving Eq. (4.20) would become a non-convex optimization problem. To address this challenge, Nguyen and Yang [159] defined gyro-structures on the SPD manifold and proposed a pseudo-gyrodistance to calculate the margin distance. Similarly, Nguyen et al. [161] proposed a pseudo-gyrodistance on the SPSD manifold based on the gyro product space. However, gyro-structures do not necessarily exist in general geometries. *In summary, the aforementioned methods often rely on specific properties of their associated Riemannian metrics, which usually do not generalize to general geometries.*

4.3.2.2 Riemannian Multinomial Logistic Regression

Recalling Eqs. (4.18) and (4.19), the minimum requirement for extending Euclidean MLR to manifolds is the well-definedness of $\text{Log}_{P_k}(S)$ for each k . In this subsection, we will develop Riemannian MLR, which depends solely on the Riemannian logarithm, without additional requirements, such as gyro-structures and the generalized law of sines. In the following, we always assume the well-definedness of the Riemannian logarithm. We start by reformulating the Euclidean margin distance to the hyperplane from a trigonometric perspective and then present our Riemannian MLR.

As we discussed before, obtaining the margin distance of Eq. (4.20) could be challenging. Inspired by Nguyen and Yang [159], we resort to the perspective of trigonometry to reinterpret Euclidean margin distance. In Euclidean space, the margin distance

is equivalent to

$$d(x, H_{a,p}) = \sin(\angle xpy^*) d(x, p), \quad \text{with } y^* = \operatorname{argmax}_{y \in H_{a,p} \setminus \{p\}} \cos \angle xpy. \quad (4.21)$$

We extend Eq. (4.21) to manifolds by the Riemannian trigonometry and geodesic distance, the counterparts of Euclidean trigonometry and distance.

Definition 112 (Riemannian margin distance). Let $\tilde{H}_{\tilde{A},P}$ be a Riemannian hyperplane defined in Eq. (4.19), and $S \in \mathcal{M}$. The Riemannian margin distance from S to $\tilde{H}_{\tilde{A},P}$ is defined as

$$d(S, \tilde{H}_{\tilde{A},P}) = \sin(\angle SPY^*) d(S, P), \quad (4.22)$$

where $d(S, P)$ is the geodesic distance, and

$$Y^* = \operatorname{argmax}_{Y \in \tilde{H}_{\tilde{A},P} \setminus \{P\}} \cos \angle SPY. \quad (4.23)$$

The initial velocities of geodesics define $\cos \angle SPY$:

$$\cos \angle SPY = \frac{\langle \operatorname{Log}_P Y, \operatorname{Log}_P S \rangle_P}{\|\operatorname{Log}_P Y\|_P \|\operatorname{Log}_P S\|_P}, \quad (4.24)$$

where $\langle \cdot, \cdot \rangle_P$ is the Riemannian metric at P , and $\|\cdot\|_P$ is the associated norm.

The Riemannian margin distance in Thm. 112 has a closed-form expression.

Theorem 113. [↓] *The Riemannian margin distance defined in Thm. 112 is given by*

$$d(S, \tilde{H}_{\tilde{A},P}) = \frac{|\langle \operatorname{Log}_P S, \tilde{A} \rangle_P|}{\|\tilde{A}\|_P}. \quad (4.25)$$

Putting Eq. (4.25) into Eq. (4.18), we can obtain a closed-form expression for Riemannian MLR.

Theorem 114 (RMLR). [↓] *Given a Riemannian manifold $(\mathcal{M}, g)$, the Riemannian MLR induced by g is*

$$p(y = k \mid S \in \mathcal{M}) \propto \exp \left(\langle \operatorname{Log}_{P_k} S, \tilde{A}_k \rangle_{P_k} \right), \quad (4.26)$$

4.3. Extension to General Riemannian Manifolds

where $P_k \in \mathcal{M}$, $\tilde{A}_k \in T_{P_k}\mathcal{M} \setminus \{\mathbf{0}\}$, and Log is the Riemannian logarithm.

As discussed for SPD MLRs in Sec. 4.2.2.2, a tangent vector attached to a trainable base point can be generated from a Euclidean parameter in a fixed tangent space through Riemannian parallel transport, vector transport, or the differential of a group translation. Following the hyperbolic and gyro MLRs [76, 159], we focus on parallel transport and Lie group translation:

$$\tilde{A}_k = \Gamma_{Q \rightarrow P_k} A_k, \quad (4.27)$$

$$\tilde{A}_k = \left(L_{P_k \odot Q_{\odot}^{-1}} \right)_{*,Q} A_k, \quad (4.28)$$

where $Q \in \mathcal{M}$ is a fixed point, $A_k \in T_Q\mathcal{M} \setminus \{\mathbf{0}\}$, Γ is the parallel transport along the geodesic connecting Q and P_k , and $\left(L_{P_k \odot Q_{\odot}^{-1}} \right)_{*,Q}$ denotes the differential map at Q of left translation $L_{P_k \odot Q_{\odot}^{-1}}$ with $P_k \odot Q_{\odot}^{-1}$ denoting the Lie group product and inverse. In this way, A_k lies in a fixed tangent space and, therefore, can be optimized by a Euclidean optimizer.

Remark 115. We make the following remarks regarding our Riemannian MLR.

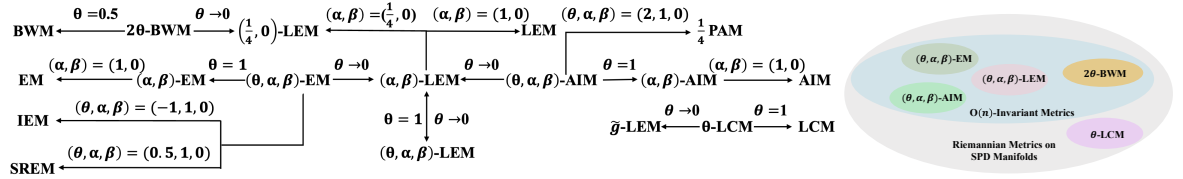
(a) The reformulations of Eq. (4.21) in gyro MLR [159, 161] and in our work are different. Gyro MLR adopts gyro trigonometry and gyro distance to reformulate Eq. (4.21), while our method directly uses Riemannian trigonometry and geodesic distance.

(b) Compared with the hyperbolic, gyro SPD, and gyro SPSD MLRs [76, 159, 161] and the SPD MLRs in Sec. 4.2, our framework enjoys broader applicability, as our framework only requires the Riemannian logarithm. This property is commonly satisfied by most manifolds encountered in machine learning, such as the five metrics on SPD manifolds mentioned in Sec. 2.9.1, the invariant metric on $\text{SO}(n)$ [28], and the hyperbolic and spherical models in Sec. 2.9.5. Besides, several existing MLRs on different geometries are special cases of our Riemannian MLR, which are detailed in Tab. 4.1.

(c) The well-definedness of the Riemannian logarithm is a much weaker requirement compared to the existence of the gyro-structure. The gyro-structure not only requires the Riemannian logarithm but also implicitly requires geodesic completeness [159, Eqs. (1)–(2)]. For instance, on SPD manifolds, EM and BWM [196] are incomplete, undermining the well-definedness of gyro operations.

MLR	Geometries	Requirements	Incorporated by Our MLR
Euclidean MLR (Eq. (4.1))	Euclidean geometry	N/A	✓(Sec. A.4.1.1)
Gyro SPD MLRs [159]	AIM, LEM & LCM on $\mathcal{S}_{++}^n$	Gyro-structures	✓(Thm. 118)
Gyro SPSP MLRs [161]	SPSP product gyro spaces	Gyro-structures	✓(Sec. A.4.1.2)
Flat SPD MLRs (Sec. 4.2)	(α, β) -LEM & (θ) -LCM on $\mathcal{S}_{++}^n$	Pullback metrics from the Euclidean space	✓(Thm. 118)
Ours	General geometries	Riemannian logarithm	N/A

Table 4.1: Several MLRs on different geometries are special cases of our MLR.


 Figure 4.2: Illustration of the deformation (left) and Venn diagram (right) of metrics on SPD manifolds, where IEM, SREM, and $\frac{1}{4}$ PAM denote Inverse Euclidean Metric, Square Root Euclidean Metric, and Polar Affine Metric scaled by $\frac{1}{4}$, respectively.

4.3.3 SPD Multinomial Logistic Regressions

This section showcases our RMLR framework on the SPD manifold. We first systematically discuss the power-deformed geometries of SPD manifolds. Based on these metrics, we will develop five families of deformed SPD MLRs.

4.3.3.1 Deformed Geometries of SPD Manifolds

As discussed in Sec. 2.9.1, there are five popular Riemannian metrics on SPD manifolds. These metrics can all be extended to power-deformed metrics. For a metric g on $\mathcal{S}_{++}^n$, the power-deformed metric is defined as

$$\tilde{g}_P(V, W) = \frac{1}{\theta^2} g_{P^\theta}((\phi_\theta)_{*,P}(V), (\phi_\theta)_{*,P}(W)), \forall P \in \mathcal{S}_{++}^n, V, W \in T_P \mathcal{S}_{++}^n, \quad (4.29)$$

where $\phi_\theta(P) = P^\theta$ is the matrix power, and $(\phi_\theta)_{*,P}$ is the differential map. The deformed metric $\tilde{g}$ can interpolate between a LEM-like metric ($\theta \rightarrow 0$) and g ($\theta = 1$) [194]. Previous work power-deformed (α, β) -AIM and BWM into (θ, α, β) -AIM [192] and 2θ -BWM [194], respectively. Earlier, we introduced (θ, α, β) -LEM and θ -LCM in Sec. 3.2.5.1. They are the power deformations of (α, β) -LEM and LCM, respectively. By Thm. 67, (θ, α, β) -LEM is equal to (α, β) -LEM. We therefore use (α, β) -LEM for this family. Here, we further define the power deformation of (α, β) -EM through Eq. (4.29),

4.3. Extension to General Riemannian Manifolds

Name	Properties
(θ, α, β) -LEM	Bi-invariance, $O(n)$ -invariance, Geodesic Completeness
(θ, α, β) -AIM	Lie Group Left-Invariance, $O(n)$ -invariance, Geodesic Completeness
(θ, α, β) -EM	$O(n)$ -Invariance
θ -LCM	Lie Group Bi-Invariance, Geodesic Completeness
2θ -BWM	$O(n)$ -Invariance

Table 4.2: Properties of deformed metrics on SPD manifolds ($\theta \neq 0$ and $\min(\alpha, \alpha + n\beta) > 0$).

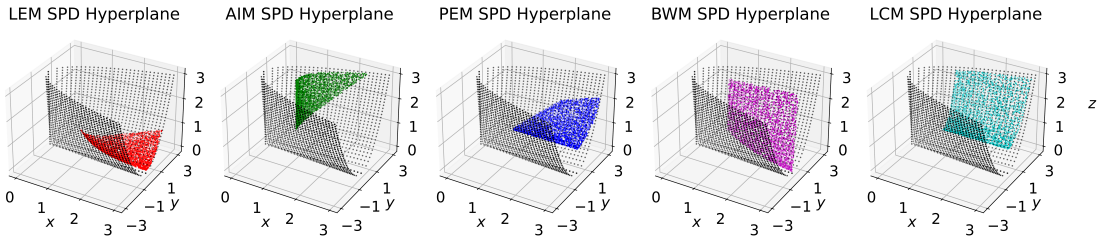


Figure 4.3: Conceptual illustration of SPD hyperplanes induced by five families of Riemannian metrics. The black dots denote the boundary of $\mathcal{S}_{++}^2$.

denoted by (θ, α, β) -EM. We have the following result for (θ, α, β) -EM.

Proposition 116. $\Downarrow$ (θ, α, β) -EM interpolates between (α, β) -LEM ($\theta \rightarrow 0$) and (α, β) -EM ($\theta = 1$).

So far, all five popular Riemannian metrics on SPD manifolds have been generalized to power-deformed families of metrics. We summarize their associated properties in Tab. 4.2 and present their theoretical relation in Fig. 4.2. We leave technical details in Sec. A.4.1.3.

4.3.3.2 Five Families of SPD Multinomial Logistic Regressions

This subsection presents five families of specific SPD MLRs using our general framework in Thm. 114 and the metrics discussed in Sec. 4.3.3.1. We focus on generating $\tilde{A}_k$ by parallel transport from the identity matrix, except for 2θ -BWM. Since the parallel transport under 2θ -BWM would undermine numerical stability (please refer to Sec. A.4.1.4 for more details), we resort to a newly developed Lie group operation [195]:

$$S_1 \odot S_2 = L_1 S_2 L_1^\top, \quad \forall S_1, S_2 \in \mathcal{S}_{++}^n, \quad (4.30)$$

where $L_1 = \text{Chol}(S_1)$ is the Cholesky factor.

Theorem 117 (SPD MLRs). $\downarrow$ By abuse of notation, we omit the subscripts k of A_k and P_k . Given an SPD feature S , the SPD MLRs, $p(y = k \mid S \in \mathcal{S}_{++}^n)$, are proportional to

$$(\alpha, \beta)\text{-LEM} : \exp [\langle \log(S) - \log(P), A \rangle^{(\alpha, \beta)}], \quad (4.31)$$

$$(\theta, \alpha, \beta)\text{-AIM} : \exp \left[\frac{1}{\theta} \left\langle \log \left(P^{-\frac{\theta}{2}} S^\theta P^{-\frac{\theta}{2}} \right), A \right\rangle^{(\alpha, \beta)} \right], \quad (4.32)$$

$$(\theta, \alpha, \beta)\text{-EM} : \exp \left[\frac{1}{\theta} \langle S^\theta - P^\theta, A \rangle^{(\alpha, \beta)} \right], \quad (4.33)$$

$$\theta\text{-LCM} : \exp \left[\frac{1}{\theta} \left\langle \begin{bmatrix} \tilde{K} \end{bmatrix} - \begin{bmatrix} \tilde{L} \end{bmatrix} + \left[\text{Dlog}(\mathbb{D}(\tilde{K})) - \text{Dlog}(\mathbb{D}(\tilde{L})) \right], \begin{bmatrix} A \end{bmatrix} + \frac{1}{2} \mathbb{D}(A) \end{bmatrix} \right\rangle, \quad (4.34)$$

$$2\theta\text{-BWM} : \exp \left[\frac{1}{4\theta} \left\langle (P^{2\theta} S^{2\theta})^{\frac{1}{2}} + (S^{2\theta} P^{2\theta})^{\frac{1}{2}} - 2P^{2\theta}, \mathcal{L}_{P^{2\theta}}(\bar{L} A \bar{L}^\top) \right\rangle \right], \quad (4.35)$$

where $A \in T_I \mathcal{S}_{++}^n \setminus \{\mathbf{0}\}$ is a symmetric matrix, $\log(\cdot)$ is the matrix logarithm, $\mathcal{L}_P[V]$ is the solution to the matrix linear system $\mathcal{L}_P[V]P + P\mathcal{L}_P[V] = V$, known as the Lyapunov operator, $\text{Dlog}(\cdot)$ is the diagonal element-wise logarithm, $\lfloor \cdot \rfloor$ is the strictly lower part of a square matrix, and $\mathbb{D}(\cdot)$ is a diagonal matrix with diagonal elements of a square matrix. Besides, $\log_{*,P}$ is the differential map at P , $\tilde{K} = \text{Chol}(S^\theta)$, $\tilde{L} = \text{Chol}(P^\theta)$, and $\bar{L} = \text{Chol}(P^{2\theta})$.

The Lyapunov operator in Eq. (4.35) requires the eigendecomposition. However, the backpropagation of eigendecomposition involves $1/(\sigma_i - \sigma_j)$ [111], undermining numerical stability. Therefore, we propose a numerically stable backpropagation for the Lyapunov operator, detailed in Sec. A.4.1.4.

As 2×2 SPD matrices can be embedded into $\mathbb{R}^3$ as an open cone [220], we illustrate SPD hyperplanes induced by five families of metrics in Fig. 4.3.

Remark 118. Our SPD MLRs extend the gyro SPD MLRs of Nguyen and Yang [159] and the flat SPD MLRs in Sec. 4.2. The pseudo-gyrodistance to an SPD hyperplane in Nguyen and Yang [159, Thms. 2.23–2.25] is incorporated by our Thm. 113, while the flat SPD MLRs under (α, β) -LEM and θ -LCM in Thm. 109 are special cases of our Thm. 117. Furthermore, our approach extends the scope of prior work because neither the framework in Sec. 4.2 nor the gyro SPD MLRs of Nguyen and Yang [159] cover SPD MLRs based on (θ, α, β) -EM and 2θ -BWM. The gyro operations in

Nguyen and Yang [159, Eq. (1)] implicitly require geodesic completeness, whereas (θ, α, β) -EM and 2θ -BWM are incomplete. As neither (θ, α, β) -EM nor 2θ -BWM belong to pullback Euclidean metrics, the framework in Thm. 108 cannot be applied to these metrics. To the best of our knowledge, our work is the **first** to apply PEM and BWM to establish Riemannian neural networks, opening up new possibilities for utilizing these metrics in machine learning applications. Besides, neither the gyro SPD MLRs of Nguyen and Yang [159] nor the flat SPD MLRs in Sec. 4.2 cover the deformed metrics for building SPD MLRs.

4.3.4 Lie Multinomial Logistic Regression

This section introduces our Lie MLR on $\text{SO}(n)$ based on the general RMLR framework in Thm. 114. The Riemannian metric on $\text{SO}(n)$ is assumed to be the invariant metric in Tab. 2.12.

We employ the vector transport on $\text{SO}(n)$ given by Boumal and Absil [28, Tab. 1], which coincides with the differential of left translation in Eq. (4.28).

Lemma 119. [↓]

$$\mathcal{T}_{Q \rightarrow P}(H) = (L_{PQ^{-1}})_{*,Q}(H) = PQ^\top H, \quad \forall P, Q \in \text{SO}(n), \quad H \in T_Q \text{SO}(n). \quad (4.36)$$

Similar to SPD MLRs, we set $Q = I$. The Lie MLR on $\text{SO}(n)$ is presented in the following.

Theorem 120. [↓] *The Lie MLR on $\text{SO}(n)$ is given by*

$$p(y = k \mid R \in \text{SO}(n)) \propto \exp(\langle \log(P_k^\top R), A_k \rangle), \quad (4.37)$$

where $P_k \in \text{SO}(n)$ and $A_k \in \mathfrak{so}(n)$.

We refer to the Riemannian hyperplanes (Eq. (4.19)) on $\text{SO}(n)$ as Lie hyperplanes. As $\text{SO}(3)$ is homeomorphic to 3-dimensional real projective space $\mathbb{RP}^3$ [95], Fig. 4.4 illustrates Lie hyperplanes in the closed ball in $\mathbb{R}^3$ of radius π .

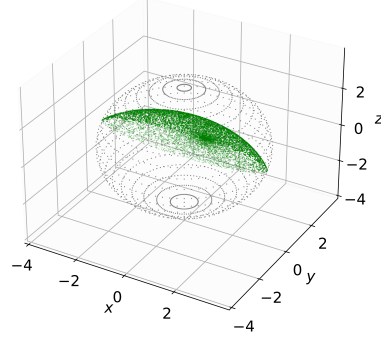


Figure 4.4: Conceptual illustration of a Lie hyperplane. Each pair of antipodal black dots corresponds to a rotation matrix with an Euler angle of π , while the green dots denote a Lie hyperplane.

4.4 Experiments

Architectures	LogEig MLR	(θ, α, β) -AIM	(θ, α, β) -EM		(α, β) -LEM		2θ -BWM		θ -LCM	
		(1,1,0)	(1,1,0)	(1,1,1/8)	(1,1,0)	(1,1,1)	(0.5)	(0.25)	(1)	(0.5)
2-Block	92.88 $\pm$ 1.05	94.53$\pm$0.95	94.24 $\pm$ 0.55	94.93$\pm$0.60	93.55 $\pm$ 1.21	95.64$\pm$0.83	92.22 $\pm$ 0.83	94.99$\pm$0.47	93.49 $\pm$ 1.25	94.59$\pm$0.82
5-Block	93.47 $\pm$ 0.45	94.32$\pm$0.94	95.11$\pm$0.82	95.01 $\pm$ 0.84	94.60 $\pm$ 0.70	95.87$\pm$0.58	93.69 $\pm$ 0.66	94.84$\pm$0.68	93.93 $\pm$ 0.98	95.16$\pm$0.67

Table 4.3: Comparison of SPDNet with LogEig against SPD MLRs on the Radar data set. The best results are **bold**.

Architectures	LogEig MLR	(θ, α, β) -AIM	(θ, α, β) -EM		(α, β) -LEM	2θ -BWM	θ -LCM	
		(1,1,0)	(1,1,0)	(0.5,1,0,1/30)	(1,1,0)	(0.5)	(1)	(0.5)
1-Block	57.42 $\pm$ 1.31	58.07 $\pm$ 0.64	66.32 $\pm$ 0.63	71.65$\pm$0.88	56.97 $\pm$ 0.61	70.24$\pm$0.92	63.84 $\pm$ 1.31	65.66$\pm$0.73
2-Block	60.69 $\pm$ 0.66	60.72 $\pm$ 0.62	66.40 $\pm$ 0.87	70.56$\pm$0.39	60.69 $\pm$ 1.02	70.46$\pm$0.71	62.61 $\pm$ 1.46	65.79$\pm$0.63
3-Block	60.76 $\pm$ 0.80	61.14 $\pm$ 0.94	66.70 $\pm$ 1.26	70.22$\pm$0.81	60.28 $\pm$ 0.91	70.20$\pm$0.91	62.33 $\pm$ 2.15	65.71$\pm$0.75

Table 4.4: Comparison of SPDNet with LogEig against SPD MLRs on the HDM05 data set.

Classifiers	LogEig MLR	(θ, α, β) -AIM		(θ, α, β) -EM	(α, β) -LEM	2θ -BWM	θ -LCM	
		(1,1,0)	(0.5,1,0.05)	(1,1,0)	(1,1,0)	(0.5)	(1)	(1.5)
Balanced Acc.	53.83±9.77	53.36±9.92	55.27±8.68	54.48±9.21	53.51±10.02	55.54±7.45	55.71±8.57	56.43±8.79

Table 4.5: Inter-session experiments of TSMNet with different MLRs on the Hinss2021 data set.

4.4. Experiments

Classifiers	LogEig MLR	(θ, α, β) -AIM		(θ, α, β) -EM		(α, β) -LEM	2θ -BWM		θ -LCM	
		(1,1,0)	(1.5,1,0)	(1,1,0)	(1.5,1,1/20)	(1,1,0)	(0.5)	(0.75)	(1)	(0.5)
Balanced Acc.	49.68 $\pm$ 7.88	50.65 $\pm$ 8.13	51.15$\pm$7.83	50.02 $\pm$ 5.81	51.38$\pm$5.77	51.41$\pm$7.98	50.26 $\pm$ 7.23	51.67$\pm$8.73	52.93 $\pm$ 7.76	54.14$\pm$8.36

Table 4.6: Inter-subject experiments of TSMNet with different MLRs on the Hinss2021 data set.

We first instantiate our SPD MLRs in four SPD neural networks: SPDNet [106] and TSMNet [123] for Riemannian feedforward networks, RResNet [117] for Riemannian residual networks, and SPDGCN [231] for Riemannian graph neural networks. Then, we proceed with experiments of our Lie MLR under the classic LieNet architecture [107]. The classifier in all the above networks is the LogEig MLR (matrix logarithm + FC + softmax), a Euclidean MLR on the tangent space at the identity matrix. We substitute the original non-intrinsic LogEig MLR in each baseline model with our RMLRs. Notably, the gyro SPD MLRs [159] are special cases of our SPD MLRs under the standard AIM, LEM, and LCM ($(\theta, \alpha, \beta) = (1, 1, 0)$), while the flat SPD MLRs in Sec. 4.2 are incorporated by our SPD MLRs under (α, β) -LEM and θ -LCM. More details on data sets and experimental settings are provided in Secs. A.1 and A.3.3.

4.4.1 Experiments on the Proposed SPD MLRs

In the following, we abbreviate *SPD MLR-metric* as *metric*. For instance, (θ, α, β) -AIM denotes the baseline endowed with the SPD MLR induced by (θ, α, β) -AIM, with $(1, 1, 0)$ as the value of (θ, α, β) .

Experiments on the Riemannian feedforward network. We evaluate our SPD MLRs for Riemannian feedforward networks under the SPDNet and TSMNet backbones. Following Huang and Van Gool [106], Brooks et al. [31], on SPDNet, we use the Radar data set [31] for radar recognition and the HDM05 data set [153] for human action recognition. TSMNet [123] is one of the state-of-the-art methods for the EEG classification task. Following Kobler et al. [123], we use the Hinss2021 [101] data set. For each family of SPD MLRs, we report the SPD MLR induced by the standard metric ($\theta = 1, \alpha = 1, \beta = 0$) and the one induced by the deformed metric with the best (θ, α, β) . Besides, if the standard SPD MLR is already saturated, we only report the results of the standard one. Under each metric, we highlight the results of our SPD MLR under the best hyperparameters in bold.

- (1) **Radar.** In line with Brooks et al. [31], we evaluate our classifiers under two network architectures: 2-Block and 5-Block configurations. The 10-fold results (mean $\pm$ std) are presented in Tab. 4.3. Note that the SPD MLR induced by standard AIM is sat-

Data Sets	LogEig MLR	(θ, α, β) -AIM	(θ, α, β) -EM	(α, β) -LEM	2θ -BWM	θ -LCM
HDM05	58.17 ± 2.07	60.23 ± 1.26	71.89 ± 0.60 ($\uparrow 13.72$)	59.44 ± 0.87	69.85 ± 0.23	65.76 ± 0.96
NTU60	45.22 ± 1.23	48.94 ± 0.68	52.24 ± 1.25	46.99 ± 0.41	50.56 ± 0.59	53.63 ± 0.95 ($\uparrow 8.41$)

Table 4.7: Comparison of LogEig against SPD MLRs under the RResNet architecture.

urated. Generally speaking, our SPD MLRs achieve superior performance against the vanilla LogEig MLR. Moreover, for most families of metrics, the associated SPD MLRs with proper (θ, α, β) outperform the standard SPD MLR, demonstrating the effectiveness of our parameterization. Besides, among all SPD MLRs, the ones induced by (α, β) -LEM achieve the best performance.

- (2) **HDM05.** Following Huang and Van Gool [106], three architectures are adopted: 1-Block, 2-Block and 3-Block configurations. The 10-fold results (mean $\pm$ std) are presented in Tab. 4.4. Note that the standard SPD MLRs under AIM, LEM, and BWM are already saturated on this data set. As on the Radar data set, similar observations can be made on this data set. Our SPD MLRs can bring consistent performance gains for SPDNet, and properly selected hyperparameters can bring further improvement. Particularly, among all the SPD MLRs, the ones based on the 2θ -BWM and (θ, α, β) -EM achieve the best performance. Compared to the vanilla LogEig MLR, **the highest performance improvement is 14.23 percentage points**, highlighting our approach’s effectiveness. Notably, since 2θ -BWM and (θ, α, β) -EM are geodesically incomplete and not pulled back from a Euclidean space, the SPD MLR under these two metrics cannot be derived by the framework of gyro or flat MLR. This contrast confirms the applicability of our theoretical framework to a broader range of geometries.
- (3) **Hinss2021.** The results (mean $\pm$ std) of leave-5%-out cross-validation are reported in Tabs. 4.5 and 4.6. Once again, our intrinsic classifiers demonstrate improved performance compared to the LogEig MLR in both inter-session and inter-subject scenarios. Besides, the SPD MLRs based on θ -LCM achieve the best performance, **outperforming the vanilla classifier by 2.60 percentage points for inter-session and by 4.46 percentage points for inter-subject**. This finding highlights the versatility of our framework.

Experiments on the Riemannian residual network. Following Katsman et al. [117], we use the HDM05 and NTU60 [178] data sets on the RResNet backbone. For the hyperparameter (θ, α, β) in our SPD MLRs, we borrow the best ones from Tab. 4.4. Tab. 4.7 reports the 10-fold and 5-fold results on the HDM05 and NTU60 data sets,

4.4. Experiments

Classifiers	Disease		Cora		Pubmed	
	Mean $\pm$ STD	Max	Mean $\pm$ STD	Max	Mean $\pm$ STD	Max
LogEig MLR	90.55 $\pm$ 4.83	96.85	78.04 $\pm$ 1.27	79.6	70.99 $\pm$ 5.12	77.6
(θ, α, β) -AIM	94.84 $\pm$ 2.27	98.43	79.79 $\pm$ 1.44	81.6	77.83 $\pm$ 1.08	80
(θ, α, β) -EM	90.87 $\pm$ 5.14	98.03	79.05 $\pm$ 1.23	81	78.16 $\pm$ 2.41	79.5
(α, β) -LEM	96.33 $\pm$ 2.19	98.82	79.89 $\pm$ 0.99	81.8	78.16 $\pm$ 2.41	79.5
2θ -BWM	91.93 $\pm$ 3.64	96.85	73.46 $\pm$ 2.18	77.7	73.22 $\pm$ 4.06	78.1
θ -LCM	93.01 $\pm$ 2.14	98.43	77.59 $\pm$ 1.20	80.1	74.46 $\pm$ 5.81	78.9

Table 4.8: Comparison of LogEig against SPD MLRs under the SPDGCN architecture.

Classifiers	Radar	HDM05	Hinss2021	
			Inter-session	Inter-subject
LogEig MLR	91.93 $\pm$ 1.30	48.43 $\pm$ 1.25	39.76 $\pm$ 7.60	44.66 $\pm$ 7.17
(θ, α, β) -AIM	95.21 $\pm$ 0.81	49.17 $\pm$ 1.08	41.14 $\pm$ 7.26	45.89 $\pm$ 6.52
(θ, α, β) -EM	92.25 $\pm$ 1.20	61.60 $\pm$ 0.69	45.78 $\pm$ 8.51 ($\uparrow$ 6.02)	45.84 $\pm$ 4.75
(α, β) -LEM	95.09 $\pm$ 0.57	49.05 $\pm$ 0.91	40.88 $\pm$ 7.46	46.02 $\pm$ 5.96 ($\uparrow$ 1.36)
2θ -BWM	94.89 $\pm$ 0.41	66.77 $\pm$ 1.34 ($\uparrow$ 18.34)	44.84 $\pm$ 8.00	45.21 $\pm$ 7.44
θ -LCM	95.67 $\pm$ 0.61 ($\uparrow$ 3.74)	58.66 $\pm$ 0.51	43.17 $\pm$ 6.21	45.10 $\pm$ 6.20

Table 4.9: Comparison of LogEig against SPD MLRs for direct classification.

respectively. The SPD MLRs still consistently outperform the vanilla LogEig MLR. Besides, similar to the SPD MLRs under the SPDNet backbone for action recognition (Tab. 4.4), the SPD MLR based on θ -LCM, 2θ -BWM, or (θ, α, β) -EM outperforms the vanilla LogEig MLR by a large margin. In particular, **the highest performance improvements are 13.72 and 8.41 percentage points** on these two data sets.

Experiments on the Riemannian graph network. We use SPDGCN [231] as the backbone network for the Riemannian graph network. Following Zhao et al. [231], we use the Disease [4], Cora [177], and Pubmed [155] data sets for node classification. The 10-fold average and maximum results of the vanilla LogEig MLR against our SPD MLR with the best (θ, α, β) are reported in Tab. 4.8. Similar to the previous results, our SPD MLRs generally outperform the LogEig MLR. Besides, the SPD MLR based on (α, β) -LEM generally achieves the best performance for SPDGCN.

Ablations of SPD MLRs on direct classification. For a more straightforward comparison, we compare LogEig against our SPD MLRs for direct classification. We adopt the Radar, HDM05, and Hinss2021 data sets. We follow the preprocessing of SPDNet and TSMNet to model features into the SPD manifold and directly use LogEig or our SPD MLRs for classification. The average results are presented in Tab. 4.9. The hyperparameters (θ, α, β) are borrowed from Tabs. 4.3 to 4.6. Our SPD MLRs consistently outperform the vanilla LogEig MLR. In particular, on the HDM05 data set, **the highest performance improvement by our SPD MLRs is 18.34 percentage points**, surpassing the non-intrinsic LogEig MLR by a large margin.

Classifiers	G3D		HDM05	
	Mean±STD	Max	Mean±STD	Max
LogEig MLR	87.91±0.90	89.73	76.92±1.27	79.11
Lie MLR	89.13±1.7	92.12	78.24±1.03	80.25

Table 4.10: Results of LogEig MLR against Lie MLR under the LieNet architecture.

4.4.2 Experiments on the Proposed Lie MLR

We apply our Lie MLR to the classic $SO(n)$ network, *i.e.*, LieNet [107], where features are on the Lie group of $SO(3) \times \cdots \times SO(3)$. More precisely, this feature space is a product manifold of $SO(3)$ factors, and Thm. 120 extends naturally under the product metric, with each class logit obtained by summing the factorwise inner products. Following LieNet [107], we use G3D [23] and HDM05 [153] data sets. We also extend the Riemannian optimization package `Geoopt` [125] to $SO(3)$, allowing for the direct Riemannian optimization reviewed in Sec. 2.7. We find that RSGD performs best for LieNet. Tab. 4.10 presents the 10-fold average results of LieNet with or without Lie MLR. Note that on the HDM05 data set, LieNet might fail to converge, with the validation accuracy fluctuating between 70% and 75%. Therefore, we select the 10 best-performing folds out of 20 experimental folds. It can be observed that our Lie MLR can improve the performance of LieNet. Besides, our Lie MLR can also improve the training stability. On the HDM05 data set, LieNet fails to converge in 8 out of 20 folds. However, when endowed with our Lie MLR, LieNet+LieMLR only encounters convergence failures in 2 folds.

4.5 Conclusion

This chapter developed a unified approach to intrinsic classification in two stages, progressing from a structured family of flat SPD geometries to general Riemannian manifolds. The first part considered SPD manifolds endowed with pullback Euclidean metrics. Their flat geometry reduces the infimum defining the geodesic distance from an SPD point to a margin hyperplane to a Euclidean point-to-hyperplane problem. This yields a closed-form margin distance and, consequently, a unified construction of SPD MLR. We instantiated this construction under deformed LEM and LCM and showed that, under the corresponding optimization scheme, its LEM instance recovers the widely used LogEig classifier, thereby providing an intrinsic interpretation of the existing pipeline.

The second part addressed the central obstacle to extending this construction beyond flat geometries. On a general Riemannian manifold, evaluating the point-to-hyperplane distance through its infimum can require solving a difficult, potentially non-convex optimization problem and may not admit a closed-form solution. Instead of solving this minimization problem, we replaced the infimum-based margin formulation with a Riemannian-trigonometric one that combines the geodesic distance from an input to the hyperplane anchor with the angle between the corresponding geodesics. This reformulation yields a closed-form RMLR that requires only a well-defined Riemannian logarithm, extending the classification principle from flat SPD geometries to a broad range of Riemannian manifolds.

We instantiated the general framework as five families of SPD MLRs under power-deformed metrics and as a Lie MLR on $\text{SO}(n)$. Experiments across Riemannian feed-forward, residual, graph, and Lie-group networks demonstrated the broad applicability of the framework.

Chapter 5

Riemannian Neural Networks

5.1 Introduction

The preceding chapters developed two fundamental network modules through unified geometric formulations that can be instantiated across different manifolds. Such unified constructions make essential modules reusable across manifold families, but not every neural component admits a sufficiently tractable or effective formulation based only on broadly shared Riemannian properties. The general RMLR in Sec. 4.3.2.2, for example, achieves broad applicability by replacing the potentially intractable point-to-hyperplane infimum with a Riemannian-trigonometric formulation that requires only a well-defined Riemannian logarithm. In contrast, hyperbolic and flat correlation geometries permit exact evaluation of the corresponding point-to-hyperplane infima, while Busemann functions and horospheres provide an alternative hyperbolic decision principle. These examples illustrate how additional geometric or algebraic structure of a particular manifold can support more direct and better-tailored modules and architectures.

This chapter therefore studies manifold-specific Riemannian network design through three complementary routes. In Sec. 5.2, we introduce the unconstrained Proper Velocity (PV) model, establish its Riemannian toolkit, and construct MLR, fully connected, convolutional, activation, and normalization layers. In Sec. 5.3, we exploit Busemann functions and horospheres to derive intrinsic and batch-efficient BMLR and BFC layers for the Poincaré and Lorentz models. Finally, Sec. 5.4 exploits the specific geometries of full-rank correlation manifolds to construct MLR, fully connected, and convolutional layers together with Riemannian backpropagation.

5.2 Proper Velocity Neural Networks

5.2.1 Introduction

Hyperbolic representations have recently delivered strong performance across different applications because the exponential volume growth of negatively curved manifolds enables low-distortion embeddings of tree-like and hierarchical structure [163]. These advantages have been validated in computer vision [78, 120, 72, 201, 79, 15, 99, 16, 189, 140, 217], graph learning [38, 13, 75, 188], multimodal learning [67, 166], recommendation systems [223], astronomy [44], genome sequence learning [119], natural language processing [163, 76, 164, 90, 98, 222], and brain signal decoding [135]. Recently, the focus has shifted from hyperbolic embeddings to building HNNs that operate entirely within hyperbolic space. As reviewed in Sec. 2.9.5, hyperbolic geometry admits multiple models, so the choice of representation is central to the design of hyperbolic networks. Most recent works rely on the Poincaré ball and hyperboloid models, which provide convenient Riemannian or gyrovector structures, thereby facilitating neural network construction. However, both models are constrained spaces, which can lead to numerical instabilities. In particular, as embeddings in the Poincaré ball approach the boundary, numerical computations become unstable and might cause gradients to vanish [91].

On the other hand, the *Proper Velocity (PV) model* originates from Einstein’s special relativity, where proper velocity provides a natural parameterization for relativistic velocity addition [200, Ch. 10]. Algebraically, PV admits a gyrovector space [200, Ch. 6], analogous to the Möbius gyrovector space of the Poincaré ball. Unlike the constrained Poincaré ball and hyperboloid models, PV offers an unconstrained representation that alleviates numerical instabilities. These properties have made the PV model successful in relativistic physics and motivate its exploration as a stable alternative geometry for HNNs. However, its Riemannian operators, including exponential and logarithmic maps and parallel transport, remain largely unexplored, despite being fundamental for constructing neural networks.

Inspired by the above discussions, we propose *Proper Velocity Neural Networks (PVNNs)*. To this end, we first establish the complete Riemannian geometry of PV by deriving closed-form expressions for the exponential map, logarithmic map, geodesic distance, and parallel transport. Building on this foundation, we extend several fundamental neural layers into PV space, including MLR classification, FC, convolutional, activation, and BN layers. Together, these layers form a complete PVNN framework

from which different network architectures can be constructed. We validate the framework through four sets of experiments, including numerical stability, image classification, graph learning, and genomic sequence learning, demonstrating both the stability of PV embeddings and effectiveness of PVNNs. To our knowledge, the PV model has remained largely unexplored in machine learning, and our work provides the first systematic study of its use for representation learning. In summary, our **contributions** are threefold:

- (1) We establish the complete Riemannian geometric toolkit of the PV manifold, deriving closed-form operators that enable its use as a new alternative to classical hyperbolic models.
- (2) We develop fundamental building blocks in PV space, including MLR, FC, convolutional, activation, and BN layers.
- (3) We validate the stability and effectiveness of PVNNs through experiments on four tasks: numerical stability, image classification, graph node classification, and genomic sequence learning.¹

Outline. In Sec. 5.2.2, we introduce the PV model and its gyrovector operations. In Sec. 5.2.3, we develop the Riemannian geometry and closed-form operators of PV space. In Sec. 5.2.4, we construct the core layers of PVNNs. In Sec. 5.2.5, we connect PV constructions to hyperboloid neural layers, and in Sec. 5.2.6, we evaluate their numerical stability and effectiveness. Proofs are deferred to Sec. B.6.

5.2.2 Preliminaries

PV Space [200]. As shown in Sec. 2.9.5, hyperbolic space is a space with constant negative curvature $K < 0$ and admits several models one can work with. The popular models include the Poincaré ball and the hyperboloid (also known as the Lorentz model). The PV model $\mathbb{PV}_K^n = \mathbb{R}^n$ is an alternative representation of hyperbolic geometry, which was initially named the Ungar gyrovector space and is used to describe algebraic structures of relativistic proper velocities [200]. Unlike the bounded Poincaré ball or the constrained hyperboloid, the PV model is an unconstrained space, offering better numerical stability. Its Riemannian metric is given by Sec. B.6.1:

$$g_x(u, v) = \langle u, v \rangle + K\beta_x^2 \langle x, u \rangle \langle x, v \rangle, \quad \forall x \in \mathbb{PV}_K^n, \forall u, v \in T_x \mathbb{PV}_K^n. \quad (5.1)$$

¹The code is available at <https://github.com/NickyoyoSu/PVNN>.

Here, $\beta_x = \frac{1}{\sqrt{1-K\|x\|^2}}$ is the relativistic beta factor. In Ungar's notation, the curvature is parametrized by a positive constant s with $s^2 = -1/K$, where s plays the role of the vacuum speed of light in special relativity [200, Sec. 3.8].

PV Gyrovector [200]. From an algebraic point of view, the PV space forms a gyrovector space [200, Def. 6.2], which extends the Euclidean vector space to manifolds. Given $x, y, z \in \mathbb{PV}_K^n$ and $t \in \mathbb{R}$, PV gyroaddition $\oplus_U$ and scalar gyromultiplication $\otimes_U$ [200, Chs. 3.11 and 6.20] are defined as²

$$x \oplus_U y = x + y + \left\{ \frac{1 - \beta_y}{\beta_y} - K \frac{\beta_x}{1 + \beta_x} \langle x, y \rangle \right\} x, \quad (5.2)$$

$$t \otimes_U y = \sinh \left(t \sinh^{-1} \left(\sqrt{-K} \|y\| \right) \right) \frac{y}{\sqrt{-K} \|y\|}, \quad (t \otimes_U \mathbf{0} = \mathbf{0}). \quad (5.3)$$

In particular, the PV inverse is $\ominus_U x = -x$, and the PV identity is the zero vector: $\mathbf{0} \oplus_U x = x \oplus_U \mathbf{0} = x$.

PV Gyration. As shown by Ungar [200, Eqs. 3.220 and 3.221], the PV gyration for any $x, y, z \in \mathbb{PV}_K^n$ is given by

$$\text{gyr}[x, y]z = z + \frac{Ax + By}{D}, \quad (5.4)$$

where the coefficients are

$$A = (1 - \beta_y^2)K \langle x, z \rangle - (1 + \beta_x)(1 + \beta_y)\beta_x\beta_yK \langle y, z \rangle + 2\beta_x^2\beta_y^2K^2 \langle x, y \rangle \langle y, z \rangle, \quad (5.5)$$

$$B = (1 - \beta_x^2)\beta_y^2K \langle y, z \rangle + (1 + \beta_x)(1 + \beta_y)\beta_x\beta_yK \langle x, z \rangle, \quad (5.6)$$

$$D = (1 + \beta_x)(1 + \beta_y) (1 - \beta_x\beta_yK \langle x, y \rangle + \beta_x\beta_y). \quad (5.7)$$

Here, $\beta_x = \frac{1}{\sqrt{1-K\|x\|^2}}$ is the relativistic beta factor.

5.2.3 Proper Velocity Geometry

5.2.3.1 From Gyro Isomorphism to Riemannian Isometry

The Poincaré ball also admits a gyrovector space, named the Möbius gyrovector space, as reviewed in Sec. 2.9.5. Algebraically, the PV and Möbius gyrovector spaces are isomorphic. We further show that PV and the Poincaré ball are geometrically isometric.

²The subscript U refers to the initial letter of Ungar.

The following bijections define the gyrovector space isomorphism [200, Tab. 6.1]:

$$\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n} : \mathbb{PV}_K^n \ni x \mapsto \frac{\beta_x}{1 + \beta_x} x \in \mathbb{P}_K^n, \quad \pi_{\mathbb{P}_K^n \rightarrow \mathbb{PV}_K^n} : \mathbb{P}_K^n \ni y \mapsto 2\gamma_y^2 y \in \mathbb{PV}_K^n, \quad (5.8)$$

where $\gamma_y = \frac{1}{\sqrt{1+K\|y\|^2}}$ is the gamma factor. The isomorphism preserves the gyro operations:

$$\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}(x \oplus_U y) = \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}(x) \oplus_M \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}(y), \quad \forall x, y \in \mathbb{PV}_K^n, \quad (5.9)$$

$$\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}(r \otimes_U x) = r \odot_M \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}(x), \quad \forall x \in \mathbb{PV}_K^n, \forall r \in \mathbb{R}, \quad (5.10)$$

where $\odot_M$ and $\oplus_M$ are the Möbius gyro operations reviewed in Sec. 2.9.5.

Lemma 121 (Differentials). [↓] *The differentials of $\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}$ and $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{PV}_K^n}$ are*

$$\begin{aligned} d_x \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}(v) &= K \frac{\beta_x^3}{(1 + \beta_x)^2} \langle x, v \rangle x + \frac{\beta_x}{1 + \beta_x} v, \quad \forall x \in \mathbb{PV}_K^n, \forall v \in T_x \mathbb{PV}_K^n, \\ d_y \pi_{\mathbb{P}_K^n \rightarrow \mathbb{PV}_K^n}(w) &= -4K\gamma_y^4 \langle y, w \rangle y + 2\gamma_y^2 w, \quad \forall y \in \mathbb{P}_K^n, \forall w \in T_y \mathbb{P}_K^n. \end{aligned}$$

Let id be the identity map. The differentials at the origin $\mathbf{0}$ are

$$d_{\mathbf{0}} \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n} = \frac{1}{2} \text{id}, \quad d_{\mathbf{0}} \pi_{\mathbb{P}_K^n \rightarrow \mathbb{PV}_K^n} = 2 \text{id}. \quad (5.11)$$

Based on Thm. 121, we can prove that the above isomorphisms are isometries.

Theorem 122 (Isometries). [↓] *The mappings in Eq. (5.8) are Riemannian isometries.*

5.2.3.2 Proper Velocity Riemannian Operators

The Poincaré ball admits the closed-form Riemannian operators reviewed in Sec. 2.9.5. By Thm. 122, we can readily obtain the counterparts on PV space via the properties of Riemannian isometries reviewed in Thm. 33.

Theorem 123 (PV Riemannian operators). $\llbracket \downarrow \rrbracket$ Let $\pi = \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}$. Given $x, y \in \mathbb{PV}_K^n$ and $v \in T_x \mathbb{PV}_K^n$, the Riemannian operators on the PV space are

$$\text{Exp}_x(v) = x \oplus_{\mathbb{U}} \left(\frac{1}{\sqrt{-K}} \sinh \left(\frac{\sqrt{-K}(1 + \beta_x)}{\beta_x} \|d_x \pi(v)\| \right) \frac{d_x \pi(v)}{\|d_x \pi(v)\|} \right), \quad (5.12)$$

$$\text{Log}_x(y) = \sigma(x, y)z + \tau(x, y) \langle x, z \rangle x, \quad (5.13)$$

$$\text{PT}_{x \rightarrow y}(v) = \frac{1 + \beta_x}{\beta_x} \tilde{v} - K \frac{(1 + \beta_x)\beta_y}{(1 + \beta_y)\beta_x} \langle y, \tilde{v} \rangle y, \quad (5.14)$$

$$d(x, y) = \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \|\pi(-x \oplus_{\mathbb{U}} y)\| \right), \quad (5.15)$$

with $z = (-x) \oplus_{\mathbb{U}} y$. For the parallel transport, $\tilde{v} = \text{gyr}_M[\bar{y}, -\bar{x}](d_x \pi(v))$ with gyr_M as the Möbius gyration in Sec. 2.9.5, $\bar{x} = \frac{\beta_x}{1 + \beta_x} x$ and $\bar{y} = \frac{\beta_y}{1 + \beta_y} y$. Here, the scalar coefficients in the logarithm are

$$\begin{aligned} \sigma(x, y) &= \frac{2}{\sqrt{-K}} \frac{\tanh^{-1}(\sqrt{-K} \|\pi(z)\|)}{\|z\|}, \\ \tau(x, y) &= \frac{2\beta_x}{1 + \beta_x} \frac{\sqrt{-K} \tanh^{-1}(\sqrt{-K} \|\pi(z)\|)}{\|z\|}. \end{aligned} \quad (5.16)$$

At the identity $\mathbf{0}$, the above operators can be further simplified:

$$\text{Exp}_{\mathbf{0}}(v) = \frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} \|v\| \right) \frac{v}{\|v\|}, \quad (5.17)$$

$$\text{Log}_{\mathbf{0}}(y) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\sqrt{-K} \|y\| \right) \frac{y}{\|y\|}, \quad (5.18)$$

$$\text{PT}_{\mathbf{0} \rightarrow y}(v) = v - K \frac{\beta_y}{1 + \beta_y} \langle y, v \rangle y, \quad (5.19)$$

$$\text{PT}_{x \rightarrow \mathbf{0}}(v) = v + K \frac{\beta_x^2}{1 + \beta_x} \langle x, v \rangle x, \quad (5.20)$$

$$d(\mathbf{0}, y) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\sqrt{-K} \|y\| \right). \quad (5.21)$$

The expressions containing normalized vectors or $\|z\|^{-1}$ are understood by continuous extension in the zero cases. Thus, $\text{Exp}_x(\mathbf{0}) = x$ and $\text{Log}_x(x) = \mathbf{0}$. In particular, $\text{Exp}_{\mathbf{0}}(\mathbf{0}) = \text{Log}_{\mathbf{0}}(\mathbf{0}) = \mathbf{0}$.

This implies that PV gyro operations can be expressed via Riemannian operations.

Theorem 124 (Gyro by Riemannian). $\llbracket \downarrow \rrbracket$ The PV gyro operations can be rewritten as

$$\begin{aligned} x \oplus_{\mathbb{U}} y &= \text{Exp}_x(\text{PT}_{\mathbf{0} \rightarrow x}(\text{Log}_{\mathbf{0}}(y))), \quad \forall x, y \in \mathbb{PV}_K^n, \\ t \otimes_{\mathbb{U}} x &= \text{Exp}_{\mathbf{0}}(t \text{Log}_{\mathbf{0}}(x)), \quad \forall x \in \mathbb{PV}_K^n, \forall t \in \mathbb{R}. \end{aligned} \quad (5.22)$$

5.2.4 Proper Velocity Neural Networks

Building on the above gyrovector and Riemannian tools, we introduce fundamental building blocks for PV neural networks, including MLR, FC, convolutional, activation, and BN layers, thereby enabling the construction of concrete deep architectures in this space.

5.2.4.1 Proper Velocity Multinomial Logistic Regression

Following the point-to-hyperplane formulation in Sec. 4.2.2.1, we define the PV margin hyperplane and solve the corresponding point-to-hyperplane infimum under the PV geometry. We define the PV hyperplane as

$$H_{a,p} = \left\{ x \in \mathbb{PV}_K^n \mid \langle \text{Log}_p(x), a \rangle_p = 0 \right\}, \quad p \in \mathbb{PV}_K^n, a \in T_p \mathbb{PV}_K^n, \quad (5.23)$$

where $p \in \mathbb{PV}_K^n$ and $a \in T_p \mathbb{PV}_K^n$ are the hyperplane parameters. As the Poincaré hyperplane can be expressed by the Möbius gyro operations [76, Eq. (22)], the PV hyperplane can also be expressed by the PV gyro operations. In addition, building PV MLR requires the PV point-to-hyperplane distance. The following theorem provides these results.

Theorem 125. $\llbracket \downarrow \rrbracket$ Let $\pi = \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{P}_K^n}$. Given $x, p \in \mathbb{PV}_K^n$ and $a \in T_p \mathbb{PV}_K^n$, we have

$$\begin{aligned} H_{a,p} &= \left\{ x \in \mathbb{PV}_K^n \mid \langle \text{Log}_p(x), a \rangle_p = 0 \right\} = \left\{ x \in \mathbb{PV}_K^n \mid \langle -p \oplus_{\mathbb{U}} x, d_p \pi(a) \rangle = 0 \right\}, \\ d(y, H_{a,p}) &= \inf_{w \in H_{a,p}} d(y, w) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{\sqrt{-K} |\langle -p \oplus_{\mathbb{U}} y, d_p \pi(a) \rangle|}{\|d_p \pi(a)\|} \right). \end{aligned}$$

By Thm. 125, we define the C -class PV MLR as

$$\begin{aligned} p(y = k \mid x) &\propto \exp(v_k(x)), \\ v_k(x) &= \text{sign}(\langle -p_k \oplus_{\mathbb{U}} x, d_{p_k} \pi(a_k) \rangle) \|a_k\|_{p_k} d(x, H_{a_k, p_k}), \end{aligned} \quad (5.24)$$

where $p_k \in \mathbb{PV}_K^n$ and $a_k \in T_{p_k} \mathbb{PV}_K^n$ are the PV MLR parameters for class k . However,

the above expression has three drawbacks: (i) the parameter p_k is over-parameterized, as it corresponds to the scalar bias parameter in the Euclidean MLR; (ii) the gyroaddition in $\langle -p_k \oplus_U x, d_{p_k} \pi(a_k) \rangle$ complicates the computation; and (iii) the parameters (p_k, a_k) are constrained, making optimization costly. To address these drawbacks, we follow Shimizu et al. [181] and adopt the parameterization $p_k = \text{Exp}_0(r_k z_k / \|z_k\|)$, $a_k = \text{PT}_{0 \rightarrow p_k}(z_k)$ with $z_k \in T_0 \mathbb{PV}_K^n \cong \mathbb{R}^n$ and $r_k \in \mathbb{R}$. This parameterization avoids Riemannian optimization in PV MLR and further simplifies the formulation.

Theorem 126 (PV MLR). $\Downarrow$ For $x \in \mathbb{PV}_K^n$, the score $v_k(x)$ in Eq. (5.24) for each class k is

$$v_k(x) = \frac{\|z_k\|}{\sqrt{-K}} \sinh^{-1} \left(\cosh(\sqrt{-K} r_k) \frac{\sqrt{-K}}{\|z_k\|} \langle x, z_k \rangle - \sinh(\sqrt{-K} r_k) \sqrt{1 - K \|x\|^2} \right), \quad (5.25)$$

where $z_k \in \mathbb{R}^n$ and $r_k \in \mathbb{R}$ are parameters for class k . In particular, as $K \rightarrow 0^-$ we have $v_k(x) \rightarrow \langle x, z_k \rangle + b_k$ with $b_k = -r_k \|z_k\|$, which recovers the Euclidean MLR reviewed in Sec. 4.2.2.1.

The parameterization (z_k, r_k) is essential for efficiency. In the original form Eq. (5.24), computing $v_k(x)$ for a batch $x \in \mathbb{R}^{b \times n}$ and C classes requires explicit gyroaddition $-p_k \oplus_U x$ for each class, producing an intermediate tensor of size $b \times C \times n$ that could cause out-of-memory errors in high dimensions. One could instead loop over classes, but this is computationally inefficient. In contrast, Eq. (5.25) depends on inner products $\langle x, z_k \rangle$, which can be implemented as a matrix multiplication.

5.2.4.2 Proper Velocity Fully Connected Layer

The Euclidean *Fully Connected (FC)* layer is defined as $y = Ax + b$ with $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. It can be expressed element-wise as $y_k = \langle a_k, x \rangle - b_k = \langle a_k, x - p_k \rangle$ with $a_k, p_k \in \mathbb{R}^n$ and $\langle p_k, a_k \rangle = b_k$. As shown by Shimizu et al. [181, Sec. 3.2] and Chen et al. [55, Sec. 3.1], the LHS y_k is the signed distance from y to the hyperplane passing through the origin and orthogonal to the k -th axis of the output space, which can be formulated as

$$\text{sign}(\langle e_k, y - 0 \rangle) d(y, H_{e_k, 0}) = \langle a_k, x - p_k \rangle, \quad \forall 1 \leq k \leq m, \quad (5.26)$$

where e_k denotes the vector whose k -th element is 1 and all others are 0.

For the PV model, the LHS of Eq. (5.26) can be formulated by the signed point-to-hyperplane distance, while the RHS can be formulated by the v_k in PV MLR. Specifi-

cally, the PV FC layer $\mathcal{F} : \mathbb{PV}_K^n \rightarrow \mathbb{PV}_K^m$ from the n -dimensional to the m -dimensional PV spaces for the input $x \in \mathbb{PV}_K^n$ returns the output $y \in \mathbb{PV}_K^m$ by solving the m equations:

$$\text{sign}(\langle d_0\pi(e_k), -\mathbf{0} \oplus_U y \rangle) d(y, H_{e_k, \mathbf{0}}) = v_k(x), \quad \forall 1 \leq k \leq m, \quad (5.27)$$

where $H_{e_k, \mathbf{0}}$ and $v_k(x)$ are given by Thm. 125 and Eq. (5.25), respectively. This definition has an explicit solution.

Theorem 127 (PV FC layer). $\downarrow$ The output $y = \mathcal{F}(x) \in \mathbb{PV}_K^m$ has the closed form

$$y_k = \frac{1}{\sqrt{-K}} \sinh(\sqrt{-K} v_k(x)), \quad 1 \leq k \leq m, \quad (5.28)$$

where $v_k(x)$ is defined in Eq. (5.25) with $z_k \in \mathbb{R}^n$ and $r_k \in \mathbb{R}$ as the FC parameters. In particular, as $K \rightarrow 0^-$ we have $y_k \rightarrow \langle x, z_k \rangle + b_k$ with $b_k = -r_k \|z_k\|$, which recovers the Euclidean FC layer.

Generalization. We can jointly express the Euclidean FC layer and activation σ , which yields the RHS of Eq. (5.26) with $\sigma(\langle a_k, x - p_k \rangle)$. Accordingly, we extend the PV FC by applying the activation to $v_k(x)$ in Eq. (5.27). Then, Eq. (5.28) becomes

$$y_k = \frac{1}{\sqrt{-K}} \sinh(\sqrt{-K} \sigma(v_k(x))), \quad 1 \leq k \leq m. \quad (5.29)$$

5.2.4.3 Proper Velocity Convolution and Activation

Convolution. As shown by Shimizu et al. [181], Bdeir et al. [15], Chen et al. [55], Euclidean convolution consists of linear maps between kernel weights and concatenated values in each receptive field. To define convolution on PV space, it therefore suffices to define PV concatenation, since we already have the PV FC layer. Because PV space is unconstrained, we define PV concatenation to coincide with Euclidean concatenation. For simplicity, we consider the 1D case. For PV inputs $\{x_i \in \mathbb{PV}_K^n\}_{i=1}^k$ in a 1D receptive field (where k is the kernel size), the PV convolution output $y \in \mathbb{PV}_K^m$ for this receptive field is $y = \mathcal{F}(\text{Concat}(x_1, \dots, x_k))$, where $\text{Concat}(\cdot)$ is standard Euclidean concatenation and $\mathcal{F}$ is the PV FC layer.

Activation. A natural choice is to apply a Euclidean activation σ in the tangent space at the origin via the mapping $x \mapsto \text{Exp}_0(\sigma(\text{Log}_0(x)))$, which has been shown to be effective in Poincaré networks [76]. Alternatively, since PV space is unconstrained, we can apply the activation directly in PV space as $x \mapsto \sigma(x)$. This direct PV-space

activation avoids exponential and logarithmic maps and is therefore more efficient.

5.2.4.4 Proper Velocity Normalization

We instantiate the GyroBN framework in Sec. 3.3.3 on the PV space and show that PV GyroBN can normalize sample statistics.

Given activations $\{x_i \in \mathbb{PV}_K^n\}_{i=1}^N$, the core operations of PV GyroBN are

$$\forall i \leq N, \quad \tilde{x}_i \leftarrow \overbrace{B \oplus_U}^{\text{Biasing}} \left(\overbrace{\frac{s}{\sqrt{v^2 + \epsilon}} \otimes_U}^{\text{Scaling}} \left(\overbrace{-M \oplus_U x_i}^{\text{Centering}} \right) \right), \quad (5.30)$$

where M and v^2 denote the Fréchet mean and variance, and $B \in \mathbb{PV}_K^n$ and $s \in \mathbb{R}$ are parameters. Owing to the isometry between the PV space and Poincaré ball, the PV Fréchet mean can be computed via the Poincaré ball: map the data to the Poincaré ball, compute the Poincaré mean [142, Alg. 1], and map the result back.

The following theorem guarantees that PV GyroBN can normalize sample statistics.

Theorem 128 (Homogeneity). $\Downarrow$ For N samples $\{x_i\}_{i=1}^N \subset \mathbb{PV}_K^n$, we have

$$\begin{aligned} \text{Homogeneity of mean: } & \text{FM}(\{B \oplus_U x_i\}_{i=1}^N) = B \oplus_U \text{FM}(\{x_i\}_{i=1}^N), \quad \forall B \in \mathbb{PV}_K^n, \\ \text{Homogeneity of dispersion from } \mathbf{0}: & \frac{1}{N} \sum_{i=1}^N d^2(t \otimes_U x_i, \mathbf{0}) = t^2 \cdot \frac{1}{N} \sum_{i=1}^N d^2(x_i, \mathbf{0}). \end{aligned}$$

Thm. 128 directly explains the PV GyroBN in Eq. (5.30). After the centering, the batch mean is shifted to the identity $\mathbf{0}$. After the scaling, the variance becomes s^2 . After the biasing, the batch mean is translated to B .

5.2.5 Connections to the Hyperboloid

This subsection discusses the connections between the PV model and the hyperboloid model. We first show the isometry between the two models. Then, we show that several current hyperboloid network layers can be rewritten as PV layers.

Proposition 129 (PV–hyperboloid isometries). $\llbracket \downarrow \rrbracket$ *The following maps are Riemannian isometries between the hyperboloid model $\mathbb{H}_K^n$ and the PV model $\mathbb{PV}_K^n$:*

$$\pi_{\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n} : \mathbb{H}_K^n \ni \begin{bmatrix} x_t \\ x_s \end{bmatrix} \mapsto x_s \in \mathbb{PV}_K^n, \quad (5.31)$$

$$\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{H}_K^n} : \mathbb{PV}_K^n \ni x \mapsto \begin{bmatrix} \sqrt{\|x\|^2 - \frac{1}{K}} \\ x \end{bmatrix} \in \mathbb{H}_K^n. \quad (5.32)$$

The PV–hyperboloid isometries in Thm. 129 imply that several standard layers in hyperboloid networks can be rewritten as PV layers composed with $\pi_{\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n}$ and $\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{H}_K^n}$.

The Lorentz activation [15, Eq. (13)], Lorentz FC layer [45, Sec. 3.1] and Lorentz concatenation [15, Eq. (32)] are

$$\text{LAct} \left(\begin{bmatrix} x_t \\ x_s \end{bmatrix} \right) = \begin{bmatrix} \sqrt{\|\sigma(x_s)\|^2 - \frac{1}{K}} \\ \sigma(x_s) \end{bmatrix}, \quad (5.33)$$

$$\text{LFC} \left(\begin{bmatrix} x_t \\ x_s \end{bmatrix} \right) = \begin{bmatrix} \sqrt{\|Wx_s + b\|^2 - \frac{1}{K}} \\ Wx_s + b \end{bmatrix}, \quad (5.34)$$

$$\text{HCat}(\{x_i\}_{i=1}^N) = \begin{bmatrix} \sqrt{\sum_{i=1}^N x_{i,t}^2 + \frac{N-1}{K}} \\ x_{1,s} \\ \vdots \\ x_{N,s} \end{bmatrix} \in \mathbb{H}_K^N, \quad (5.35)$$

where $x = [x_t, x_s^\top]^\top \in \mathbb{H}_K^n$ and $x_i = [x_{i,t}, x_{i,s}^\top]^\top \in \mathbb{H}_K^n$ for $1 \leq i \leq N$. Then Thm. 129 implies that the above Lorentz layers can be rewritten in terms of PV layers as follows:

$$\text{LAct}(x) = \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{H}_K^n}(\sigma(\pi_{\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n}(x))), \quad (5.36)$$

$$\text{LFC}(x) = \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{H}_K^n}(W\pi_{\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n}(x) + b), \quad (5.37)$$

$$\text{HCat}(\{x_i\}_{i=1}^N) = \pi_{\mathbb{PV}_K^n \rightarrow \mathbb{H}_K^n}(\text{Concat}(\pi_{\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n}(x_1), \dots, \pi_{\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n}(x_N))). \quad (5.38)$$

These identities show that many hyperboloid constructions effectively operate by mapping to PV space, applying Euclidean building blocks there, and mapping back through $\pi_{\mathbb{PV}_K^n \rightarrow \mathbb{H}_K^n}$. This perspective naturally motivates designing networks directly in PV space, instead of repeatedly switching between equivalent models. Moreover, even if one

follows the pattern $\mathbb{H}_K^n \rightarrow \mathbb{PV}_K^n \rightarrow \mathbb{PV}_K^m \rightarrow \mathbb{H}_K^m$ to construct layers, the intermediate map should be the PV layers, such as Thm. 127, rather than Euclidean layers, since PV is a non-linear Riemannian manifold.

5.2.6 Experiments

We evaluate PV embeddings and PVNNs on four representative tasks:

- Sec. 5.2.6.1 evaluates the numerical advantage of the PV model against Poincaré and hyperboloid.
- Sec. 5.2.6.2 compares PV, Poincaré, and hyperboloid MLRs on image classification.
- Sec. 5.2.6.3 evaluates our PV MLR, FC, and GyroBN layers on graph learning.
- Sec. 5.2.6.4 compares fully PV convolutional networks with fully hyperboloid convolutional networks on genomic sequence learning.

More details on data sets and experimental settings are provided in Secs. A.1 and A.3.4.

5.2.6.1 Numerical Stability

We study three aspects: gyro operator, Riemannian operator, and gradient behavior. All experiments use curvature $K = -1$, dimension $n = 16$, and batch size 4096.

Gyro Operator. We use scalar gyromultiplication $r \otimes_{\mathcal{H}} x$ as a probe of numerical stability across hyperbolic models. Given random batches x and radii r , we evaluate two metrics. The *failure*

rate is the fraction of outputs that contain NaN/Inf. The *violation rate* is defined only for models with manifold constraints:

Poincaré ball requires $\|x\|^2 < -1/K$, and hyperboloid requires $\langle x, x \rangle_{\mathcal{L}} = \|x_s\|^2 - x_t^2 = \frac{1}{K}$ for $x = [x_t, x_s^\top]^\top$. The tolerance is set to 10^{-8} . As PV is unconstrained, its violation rate is reported as N/A. As shown in Tab. 5.1, PV maintains zero failures up to $r = 1000$ in

r	Failure rate			Violation rate		
	$\mathbb{PV}_K^n$	$\mathbb{P}_K^n$	$\mathbb{H}_K^n$	$\mathbb{PV}_K^n$	$\mathbb{P}_K^n$	$\mathbb{H}_K^n$
1	0	0	0	N/A	0	32.50
5	0	0	0	N/A	0	92.36
10	0	0	0	N/A	0	99.76
20	0	0	4.23	N/A	0	100
50	0	0	64.42	N/A	0	100
75	0	0	79.63	N/A	0	100
100	0	0	88.26	N/A	0	100
150	0	0	96.43	N/A	0	100
200	0	0	100	N/A	0	100
1000	0	0	100	N/A	0	100

Table 5.1: Failure and violation rates (%) of $r \otimes_{\mathcal{H}} x$ in FP32.

FP32. The Poincaré ball has zero failure and violation rates, whereas the hyperboloid model starts to fail around $r = 20$ and quickly accumulates both NaN/Inf outputs and off-manifold points under large scalar multipliers, revealing pronounced numerical instability.

Riemannian Operator. We evaluate the exponential and logarithmic maps by measuring the round-trip error $\|\text{Log}_0(\text{Exp}_0(v)) - v\|$ for tangent vectors v with large norm $\|v\| = 10$. Since this quantity is theoretically zero, any non-zero value reflects numerical instability. We sample a batch of such vectors and report the average error in

Model	FP32	FP64
$\mathbb{P}_K^n$	2.1×10^{-4}	4.3×10^{-11}
$\mathbb{H}_K^n$	1.0×10^0	1.0×10^0
$\mathbb{PV}_K^n$	2.1×10^{-7}	6.7×10^{-16}

Table 5.2: $\|\text{Log}_0(\text{Exp}_0(v)) - v\|$.

Tab. 5.2. PV achieves stable behavior in both FP32 and FP64, whereas the Poincaré ball already exhibits noticeable errors in FP32 and the hyperboloid model remains unstable in both precisions.

Gradient. To compare gradient behavior, we study the gradient of $f_r(x) = \|r \otimes_{\mathcal{H}} x - x\|$ with respect to x . Specifically, we sample 24 logarithmically spaced radii $r \in [1, 1000]$ and, for each radius, measure the $\|\nabla_x f_r(x)\|$ on a random batch.

Model	$\ \nabla_x f_r(x)\ $ Range	Gradient behavior
$\mathbb{P}_K^n$	$[7.6 \times 10^{-13}, 1.1 \times 10^{-11}]$	Vanishing gradients
$\mathbb{H}_K^n$	$[0, \text{NaN}]$	Exploding gradients
$\mathbb{PV}_K^n$	$[2.1 \times 10^{-6}, 1.1 \times 10^{-4}]$	Stable gradients

Table 5.3: Gradient magnitude $\|\nabla_x f_r(x)\|$ across varying radii.

The range of $\|\nabla_x f_r(x)\|$ is summarized in Tab. 5.3. The Poincaré ball exhibits severe gradient vanishing near the boundary. In contrast, the hyperboloid model yields gradients that vary from 0 to NaN, reflecting gradient explosion. PV maintains gradients in a safer band.

5.2.6.2 Image Classification

We compare our PV MLR against previous Poincaré MLRs [76, 181] and Lorentz MLR [15]. Following Bdeir et al. [15], we train a ResNet-18 backbone [97] on CIFAR-10 and CIFAR-100 [126], replacing the final Euclidean MLR with a hyperbolic MLR. The backbone output is lifted to the target geometry via the exponential map at the identity. Since PV space is unconstrained, we also consider a direct variant that skips Exp_0 and treats the backbone output as PV coordinates. We denote these two PV heads as PV MLR (with Exp_0) and PV MLR (without Exp_0). Tab. 5.4 reports the 5-fold results.

5.2. Proper Velocity Neural Networks

Model	Method	CIFAR-10 ($\delta = 0.26$)	CIFAR-100 ($\delta = 0.23$)
$\mathbb{P}_K^n$	Poincaré MLR [76, Eq. (25)]	95.09 ± 1.51	76.78 ± 0.67
	Unidirectional MLR [181, Eq. (6)]	95.12 ± 0.20	77.19 ± 0.10
$\mathbb{H}_K^n$	Lorentz MLR [15, Thm. 2]	95.02 ± 0.12	77.96 ± 0.09
$\mathbb{P}\mathbb{V}_K^n$	PV MLR (with Exp_0)	95.27 ± 0.12	78.19 ± 0.59
	PV MLR (without Exp_0)	95.30 ± 0.18	78.20 ± 0.37

Table 5.4: Top-1 image classification accuracy (%) of hyperbolic MLRs on ResNet-18. The best results are **bold**. δ represents the δ -hyperbolicity (lower is more hyperbolic), which comes from Bdeir et al. [15, Tab. 1].

Model	Method	Disease ($\delta = 0$)	Airport ($\delta = 1$)	PubMed ($\delta = 3.5$)	Cora ($\delta = 11$)
$\mathbb{K}_K^n$	KNN [147]	79.41 ± 0.55	92.10 ± 0.97	69.36 ± 0.76	52.26 ± 1.99
$\mathbb{P}_K^n$	HNN [76]	79.90 ± 0.01	82.16 ± 2.95	69.28 ± 0.85	49.68 ± 1.25
	HNN++ [181]	80.57 ± 0.23	88.40 ± 0.17	73.68 ± 0.39	52.06 ± 0.90
$\mathbb{H}_K^n$	LNN [15]	79.90 ± 0.01	75.20 ± 1.08	68.82 ± 0.88	53.34 ± 1.65
$\mathbb{P}\mathbb{V}_K^n$	PVNN	81.15 ± 0.23	97.96 ± 0.42	74.33 ± 0.22	51.42 ± 1.33

Table 5.5: Accuracies of hyperbolic networks on graph learning. The best results are **bold**. δ represents the δ -hyperbolicity (lower is more hyperbolic).

PV MLR matches or outperforms prior hyperbolic baselines, with the largest gains on CIFAR-100 where the decision boundaries are more complex. Both PV variants, with and without Exp_0 , achieve similar accuracies.

5.2.6.3 Graph Learning

Data and Setup. We study node classification on four standard graph data sets: Disease [4], Airport [229], Cora [177], and PubMed [155]. All models share the same architecture consisting of two FC layers with nonlinear activations followed by an MLR classifier; they differ only in the underlying hyperbolic model. Baselines include KNN [147] for the Klein ball, HNN/HNN++ [38, 181] for the Poincaré ball, and LNN [15] for the hyperboloid model. Our PVNN is built from PV FC, activation, and MLR layers.

Main Results. For a fair comparison, we use a tangent activation in each model and set $\sigma = \text{id}$ for the PV FC layer in Eq. (5.29). Tab. 5.5 summarizes the 5-fold results. On the three more hyperbolic data sets (Disease, Airport, and PubMed), PVNN consistently achieves the best performance, with especially large gains on Airport where it improves over the strongest baseline by 5.86%. On the weakly hyperbolic Cora data set, PVNN remains comparable to Poincaré- and Klein-based networks, and worse than

Method	Disease	Airport	PubMed	Cora
PVNN+TFC	80.86 $\pm$ 0.30	86.99 $\pm$ 0.61	74.40 $\pm$ 0.43	53.58 $\pm$ 0.81
PVNN	81.24 $\pm$ 0.36	97.93 $\pm$ 0.29	74.16 $\pm$ 0.32	52.26 $\pm$ 1.32
PVNN+TBN	80.67 $\pm$ 0.38	98.71 $\pm$ 0.36	73.52 $\pm$ 0.12	45.36 $\pm$ 2.44
PVNN+GyroBN	81.24 $\pm$ 0.19	99.03 $\pm$ 0.18	74.34 $\pm$ 0.31	46.64 $\pm$ 5.45

Table 5.6: Results of Tangent FC (TFC) vs PV FC, and Tangent BN (TBN) vs GyroBN.

Method	Disease		Airport		PubMed		Cora	
	Acc	Fit Time	Acc	Fit Time	Acc	Fit Time	Acc	Fit Time
Tangent	81.15 $\pm$ 0.23	26.08	98.56 $\pm$ 0.36	55.48	61.50 $\pm$ 5.75	3.10	33.10 $\pm$ 1.58	7.12
Euclidean	81.15 $\pm$ 0.23	25.80	98.75 $\pm$ 0.31	55.19	69.82 $\pm$ 3.58	2.99	32.62 $\pm$ 0.65	7.29
Fréchet 1 iter	81.05 $\pm$ 0.23	29.79	88.93 $\pm$ 1.17	65.19	62.52 $\pm$ 8.44	3.38	42.84 $\pm$ 6.15	7.67
Fréchet 2 iters	81.05 $\pm$ 0.23	30.12	94.11 $\pm$ 0.46	67.37	73.78 $\pm$ 0.20	3.49	45.68 $\pm$ 4.36	8.21
Fréchet 5 iters	81.24 $\pm$ 0.36	30.90	98.50 $\pm$ 0.16	82.28	73.92 $\pm$ 0.44	4.02	49.50 $\pm$ 1.83	9.15
Fréchet 10 iters	81.24 $\pm$ 0.19	30.49	99.03 $\pm$ 0.18	105.79	74.34 $\pm$ 0.31	3.96	46.64 $\pm$ 5.45	9.77
Fréchet ∞	80.86 $\pm$ 0.00	31.29	98.46 $\pm$ 0.15	122.37	71.16 $\pm$ 3.93	4.46	47.32 $\pm$ 4.73	9.27

Table 5.7: Comparison of methods in calculating mean and variance in PV GyroBN. Time is measured in milliseconds per training epoch.

the hyperboloid-based one. Overall, these results suggest that PV geometry is more effective on strongly hyperbolic graphs.

Tangent vs. Riemannian. A natural construction of hyperbolic layers is to work in the tangent space. To validate the benefits of our Riemannian PV layers, we compare our PV FC with TFC of the form $\text{Exp}_{\mathbf{0}}(A \text{Log}_{\mathbf{0}}(x) + b)$, and our GyroBN with TBN given by $\text{Exp}_{\mathbf{0}}(\text{BN}(\text{Log}_{\mathbf{0}}(x)))$ [109]. We denote these variants by PVNN+TFC and PVNN+TBN, respectively. As shown in Tab. 5.6, PVNN consistently outperforms PVNN+TFC on the more hyperbolic Disease and Airport data sets, while performance on the other two data sets is comparable and TFC can be slightly better. For normalization, PVNN+GyroBN improves over PVNN+TBN on all data sets. Overall, these ablations validate the effectiveness of our Riemannian PV constructions, especially in strongly hyperbolic settings.

Ablations on Batch Statistics. PV GyroBN in Eq. (5.30) uses Fréchet mean and variance, which require iterative solvers. We also consider two efficient variants. A tangent variant computes batch statistics in the tangent space at the identity via

$$\begin{aligned}
 M &= \text{Exp}_{\mathbf{0}} \left(\frac{1}{N} \sum_{i=1}^N \text{Log}_{\mathbf{0}}(x_i) \right), \\
 v^2 &= \frac{1}{N} \sum_{i=1}^N \|\text{Log}_{\mathbf{0}}(x_i) - \text{Log}_{\mathbf{0}}(M)\|^2,
 \end{aligned} \tag{5.39}$$

and a Euclidean variant computes standard Euclidean mean and variance directly in

5.2. Proper Velocity Neural Networks

Exp ₀	Disease	Airport	PubMed	Cora
✗	81.05 ± 0.23	97.71 ± 0.34	74.22 ± 0.26	51.92 ± 2.01
✓	81.24 ± 0.36	97.93 ± 0.29	74.16 ± 0.32	52.26 ± 1.32

Table 5.8: Ablations on PVNN with or without exponential map for the input PV feature.

Method	Disease	Airport	PubMed	Cora
Tangent Act.	81.24 ± 0.36	97.93 ± 0.29	74.16 ± 0.32	52.26 ± 1.32
FC σ	81.34 ± 0.43	99.40 ± 0.15	74.02 ± 0.17	51.34 ± 0.46
FC σ + Tangent Act.	80.96 ± 0.19	99.15 ± 0.38	73.96 ± 0.22	51.30 ± 1.65
Euc. Act.	81.34 ± 0.43	98.87 ± 0.35	74.56 ± 0.59	38.10 ± 3.30

Table 5.9: Ablations on PV activations.

the unconstrained PV space. Tab. 5.7 shows that Tangent and Euclidean are up to $2\times$ faster while achieving similar accuracies on Disease and Airport. Although Fréchet-based GyroBN attains the best accuracies, it is more computationally expensive.

Ablations on PV Embedding. In the main experiments, the input features are first lifted to PV via Exp₀ and then processed by PVNN. Since PV space is unconstrained, we also consider a variant that feeds the Euclidean features directly as PV coordinates. Tab. 5.8 compares these two settings. The two variants perform similarly, while using Exp₀ provides small improvements on Disease, Airport, and Cora. This differs from image classification in Tab. 5.4, where the variant without Exp₀ is marginally better. This slight discrepancy may stem from the different nature of the inputs. In vision, the ResNet encoder can adapt its learned representation to the chosen lifting, whereas in graphs the raw node features benefit slightly from the explicit exponential map.

Ablations on Activation. We ablate three types of activations in PVNN: the internal nonlinearity σ in the PV FC layer (fixed to tanh), and explicit activations applied either directly in PV (*Euc. Act.*) or in the tangent space (*Tangent Act.*). Tab. 5.9 reports the results. First, when comparing these three choices individually, the differences are small on Disease and PubMed, while FC σ performs best on Airport, Tangent Act. performs best on Cora, and Euc. Act. degrades substantially on Cora. Second, when comparing the composite variant FC σ + Tangent Act. against Tangent Act., the composite does not yield consistent gains, suggesting redundancy.

Task	Data Set	Euclidean CNN	HCNN-S	PVCNN
Retrotransposons	LINEs	70.63 ± 1.24	76.12 ± 2.16	81.83 ± 0.27
	SINEs	85.15 ± 1.64	85.45 ± 1.16	93.78 ± 0.54
DNA transposons	hAT-Ac	87.45 ± 0.90	89.61 ± 1.34	92.08 ± 0.80
Pseudogenes	processed	60.66 ± 0.82	68.30 ± 0.93	71.27 ± 0.78
	unprocessed	51.94 ± 2.69	56.10 ± 0.56	62.31 ± 0.78

Table 5.10: Comparison in MCC of hyperbolic and Euclidean convolutional networks, including PVCNN, on TEB data sets.

5.2.6.4 Genomic Sequence Learning

Khan et al. [119] recently proposed *Hyperbolic Convolutional Neural Networks (HCNNs)* on the hyperboloid for genomic sequence learning, demonstrating that HCNNs outperform Euclidean CNNs on this task. Following Khan et al. [119], we evaluate on the *Transposable Elements Benchmark (TEB)* data set for DNA transposable element prediction. To ensure a fair comparison, all models share the same backbone network architecture, which consists of two convolutional blocks followed by an FC layer and a final MLR classifier [119]. We use a single curvature shared for all layers. Tab. 5.10 reports 5-fold *Matthews Correlation Coefficient (MCC)*. The *PV Convolutional Network (PVCNN)* achieves the best performance on all TEB tasks, with particularly strong gains on SINEs, where it improves over HCNN-S by about 9 MCC points. These results demonstrate the benefits of PV convolutional networks.

5.3 Hyperbolic Busemann Neural Networks

5.3.1 Introduction

The preceding section introduced the unconstrained PV representation and its neural layers, based on geodesic hyperplanes. We next use another powerful geometric tool, the Busemann function, to construct hyperbolic neural networks. For simplicity, we focus on the widely used Poincaré ball and Lorentz models.

To support deep learning fully in hyperbolic spaces, several key building blocks in neural networks have recently been generalized to Poincaré or Lorentz spaces, including attention [90, 45, 221], BN [142, 15, 52, 54], linear feed-forward layers [76, 181, 45], activation [76, 15], residual blocks [201, 118, 99], MLR [181, 15, 162], and graph convolution [38, 139, 13, 60]. Among these components, MLR classification and FC layers play a fundamental role in final decision-making and feature transformation.

Recently, hyperplanes and point-to-hyperplane distances, which have been explored in Chapter 4 and Sec. 5.2, have been adopted to construct hyperbolic MLR in both Poincaré [76, 181] and Lorentz [15] models. Ganea et al. [76, Sec. 3.1] introduced the first Poincaré MLR based on Poincaré hyperplanes, but the formulation suffers from over-parameterization and lacks batch efficiency. Shimizu et al. [181, Sec. 3.1] alleviated these issues through a re-parameterization strategy. Building on these ideas, Bdeir et al. [15, Sec. 4.3] proposed a Lorentz MLR. However, its hyperplane is defined by the ambient Minkowski space, which is model-specific and may distort Lorentzian geometry.

For hyperbolic FC layers, three main formulations exist. Ganea et al. [76, Sec. 3.2] introduced Möbius matrix-vector multiplication through the tangent space on the Poincaré ball. Shimizu et al. [181, Sec. 3.2] further proposed the Poincaré FC layer, defined intrinsically but restricted to the Poincaré model. On the Lorentz model, Chen et al. [45, Sec. 2.2] constructed a Lorentz FC layer by applying linear transformations in the ambient Minkowski space followed by projection onto the Lorentz model. Thus, Möbius and Lorentz FC rely on flat-space (tangent or ambient) approximations that could distort intrinsic geometry, whereas Poincaré FC is intrinsic but model-specific.

On the other hand, the Busemann function and its level sets, horospheres, have emerged as powerful intrinsic tools for hyperbolic learning. They enjoy convenient metric properties [29, Ch. II.8] and admit closed-form expressions on both the Poincaré and Lorentz models [24, Prop. 9]. These operators have supported several hyperbolic algorithms, including SVM [73], PCA [39], Sliced Wasserstein distances [24], and prototype learning [81]. We also note that Nguyen et al. [162, Cor. 4.3] proposed a Poincaré MLR based on the Busemann function. However, its induced point-to-hyperplane distance is pseudo, coincides with the true distance only in Euclidean geometry, remains over-parameterized, and is not batch efficient.

These observations motivate intrinsic and batch-efficient formulations for MLR and FC layers that can operate on both the Poincaré ball and the Lorentz model. To address this need, we propose *Busemann Multinomial Logistic Regression (BMLR)* and *Busemann Fully Connected (BFC)* layers, two Busemann-based components for hyperbolic networks. Our contributions are summarized as follows:

- We introduce BMLR, deriving intrinsic logits directly from Busemann functions with a point-to-horosphere distance interpretation. BMLR uses a compact per-class parameterization, eliminates manifold-valued parameters in prior MLRs, remains batch-efficient, and recovers Euclidean MLR as curvature tends to zero.

- We develop BFC layers by generalizing the FC and activation layers through the Busemann function, providing intrinsic constructions on both the Poincaré and Lorentz models. BFC preserves comparable complexity and parameter counts, and recovers Euclidean FC in the zero curvature limit.
- We provide empirical validation across image classification, genome sequence learning, node classification, and link prediction. BMLR and BFC generally outperform existing hyperbolic layers. BMLR shows particularly large gains as the number of classes increases, and the Lorentz BMLR is the fastest among all hyperbolic MLRs.³

Outline. In Sec. 5.3.2, we recall Busemann functions and horospheres in hyperbolic space. In Sec. 5.3.3, we introduce BMLR and its point-to-horosphere interpretation. In Sec. 5.3.4, we develop BFC layers, and in Sec. 5.3.5, we evaluate both components. Proofs are deferred to Sec. B.7.

5.3.2 Preliminaries

The metric-geometric notions of geodesic rays, asymptotic rays, Busemann functions, horoballs, horospheres, and Hadamard spaces have been reviewed in Sec. 2.5. The Poincaré and Lorentz models and their Riemannian operators have been reviewed in Sec. 2.9.5. Their gyro operators are presented in Tab. 2.13 and Sec. 3.3.4.5, respectively. Their gyrovector spaces are denoted by $\{\mathbb{P}_K^n, \oplus_M, \odot_M\}$ and $\{\mathbb{L}_K^n, \oplus_L, \odot_L\}$, respectively. In the following, we review the Busemann function on the hyperbolic space.

In Euclidean space, the Busemann function associated with the geodesic $\gamma(t) = tv$ that starts at $\mathbf{0}$ with unit direction $v \in \mathbb{S}^{n-1}$ is $B^v(x) = -\langle x, v \rangle$, which coincides, up to a sign, with the inner product. Let $\mathcal{H}_K^n \in \{\mathbb{P}_K^n, \mathbb{L}_K^n\}$ be a hyperbolic space. We write $B^v(x)$ for the Busemann function associated with the ray that emanates from the origin $e \in \mathcal{H}_K^n$ in the direction $v \in \mathbb{S}^{n-1} \subset T_e \mathcal{H}_K^n$. For $K < 0$, $v \in \mathbb{S}^{n-1}$, and $x \in \mathcal{H}_K^n$, closed forms of the Poincaré and Lorentz Busemann functions [24, Prop. 9] are

$$\mathbb{P}_K^n : \quad B^v(x) = \frac{1}{\sqrt{-K}} \log \left(\frac{\|v - \sqrt{-K}x\|^2}{1 + K\|x\|^2} \right), \quad (5.40)$$

$$\mathbb{L}_K^n : \quad B^v(x) = \frac{1}{\sqrt{-K}} \log \left(\sqrt{-K} (x_t - \langle x_s, v \rangle) \right). \quad (5.41)$$

³The code is available at <https://github.com/GitZH-Chen/HBNN>.

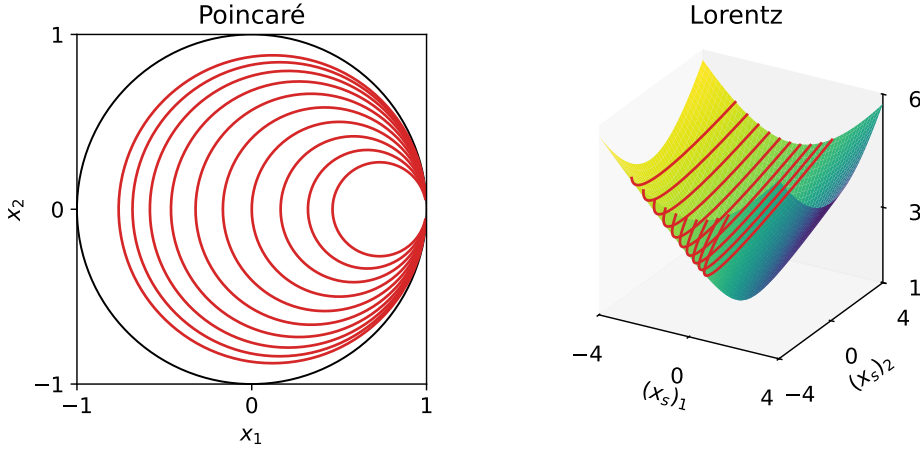


Figure 5.1: Illustration: red curves are different horospheres of B^v .

The level sets of a Busemann function are horospheres, the hyperbolic counterpart of Euclidean hyperplanes. In Euclidean space, for a unit direction v , the hyperplanes $H_\tau^v = \{x \in \mathbb{R}^n \mid \langle x, v \rangle = \tau\}$ with $\tau \in \mathbb{R}$ are parallel. Analogously, for fixed v , the horospheres $H_\tau^v = \{x \in \mathcal{H}_K^n \mid B^v(x) = \tau\}$ are equidistant, as established later in Thm. 132. Fig. 5.1 illustrates such horospheres, and Tab. 2.4 summarizes the above correspondence between Euclidean and hyperbolic notions.

5.3.3 Busemann Multinomial Logistic Regression

We begin by reformulating the Euclidean MLR, then lift it to hyperbolic space via the Busemann function, introducing BMLR. We also present a point-to-horosphere interpretation. Finally, we compare BMLR with existing hyperbolic MLRs, highlighting our advantages in geometric fidelity, parameterization, and computational efficiency.

5.3.3.1 Formulation

The Euclidean MLR softmax($Ax + b$) computes the multinomial probability for each class $k \in \{1, \dots, C\}$ given an input $x \in \mathbb{R}^n$. It admits the inner product form:

$$\forall k, \quad p(y = k \mid x) = \frac{\exp(\langle a_k, x \rangle + b_k)}{\sum_{j=1}^C \exp(\langle a_j, x \rangle + b_j)}, \quad (5.42)$$

where $a_k \in \mathbb{R}^n$ and $b_k \in \mathbb{R}$ are the weight and bias for class k . We write $p(y = k \mid x) \propto \exp(u_k(x))$ with $u_k(x) = \langle a_k, x \rangle + b_k$. Decomposing the weight vector into a magnitude

$\alpha_k = \|a_k\| > 0$ and a unit direction $v_k = \frac{a_k}{\|a_k\|} \in \mathbb{S}^{n-1}$, each logit is

$$u_k(x) = \alpha_k \langle v_k, x \rangle + b_k. \quad (5.43)$$

As reviewed in Sec. 5.3.2, the Busemann function naturally generalizes the Euclidean inner product. Analogously to Eq. (5.43), we define the hyperbolic logits via the Busemann function, yielding BMLR:

$$\forall k, \quad p(y = k \mid x) = \frac{\exp(u_k(x))}{\sum_{j=1}^C \exp(u_j(x))}, \quad (5.44)$$

$$u_k(x) = -\alpha_k B^{v_k}(x) + b_k, \quad (5.45)$$

with $\alpha_k > 0$, $v_k \in \mathbb{S}^{n-1}$, and $b_k \in \mathbb{R}$ as parameters.

The following result shows that, as $K \rightarrow 0^-$, both the Poincaré and Lorentz BMLRs reduce to the Euclidean MLR.

Theorem 130 (Limits of BMLRs). $\llbracket \downarrow \rrbracket$ As $K \rightarrow 0^-$, the hyperbolic Busemann functions converge to the Euclidean inner product:

$$\text{Poincaré: } B^v(x) \xrightarrow{K \rightarrow 0^-} -2 \langle v, x \rangle, \quad (5.46)$$

$$\text{Lorentz: } B^v(x) \xrightarrow{K \rightarrow 0^-} -\langle v, x_s \rangle. \quad (5.47)$$

The hyperbolic BMLRs converge to the Euclidean MLR:

$$\text{Poincaré: } u_k(x) \xrightarrow{K \rightarrow 0^-} 2\alpha_k \langle v_k, x \rangle + b_k, \quad (5.48)$$

$$\text{Lorentz: } u_k(x) \xrightarrow{K \rightarrow 0^-} \alpha_k \langle v_k, x_s \rangle + b_k. \quad (5.49)$$

Remark 131 (Intuition). On the Poincaré ball, letting $K \rightarrow 0^-$ recovers Euclidean geometry [182, App. A.4.2]. For the Lorentz model, as $K \rightarrow 0^-$, the temporal coordinate diverges while the spatial component approaches $\mathbb{R}^n$, making $\mathbb{L}_K^n$ converge to a Euclidean space. Consistently, the Poincaré and Lorentz Busemann functions and the associated BMLR logits reduce to their Euclidean counterparts, providing a natural generalization of Euclidean MLR.

5.3.3.2 Geometric Interpretation

The point-to-hyperplane strategy underlying MLR has been established in Sec. 4.2.2.1. We now show that BMLR admits the corresponding interpretation through point-to-horosphere distances.

Theorem 132 (Hadamard horosphere distance). $\Downarrow$ Let $(\mathcal{X}, d)$ be a geodesically complete Hadamard space, and let $B^\gamma : \mathcal{X} \rightarrow \mathbb{R}$ be the Busemann function associated with a geodesic ray $\gamma : [0, \infty) \rightarrow \mathcal{X}$. For any $\tau_1, \tau_2 \in \mathbb{R}$, define the horospheres by

$$H_{\tau_i}^\gamma = \{x \in \mathcal{X} \mid B^\gamma(x) = \tau_i\}, \quad i = 1, 2. \quad (5.50)$$

The distance between these horospheres is constant:

$$d(H_{\tau_1}^\gamma, H_{\tau_2}^\gamma) = d(H_{\tau_2}^\gamma, H_{\tau_1}^\gamma) = |\tau_2 - \tau_1|. \quad (5.51)$$

In particular, the point-to-horosphere distance is

$$d(x, H_\tau^\gamma) = |B^\gamma(x) - \tau|, \quad \forall x \in \mathcal{X}. \quad (5.52)$$

Corollary 133 (Point-to-horosphere distance). In a hyperbolic space

$$\mathcal{H}_K^n \in \{\mathbb{P}_K^n, \mathbb{L}_K^n\} \quad (5.53)$$

with curvature $K < 0$, the point-to-horosphere distance is

$$d(x, H_\tau^v) = |B^v(x) - \tau|, \quad (5.54)$$

where $H_\tau^v = \{x \mid B^v(x) = \tau\}$ denotes the horosphere with respect to direction $v \in \mathbb{S}^{n-1}$.

A Euclidean hyperplane can be parameterized by a unit direction, a positive magnitude, and a scalar bias. Similarly, we parameterize a hyperbolic horosphere as

$$H_{v, \alpha, b} = \{x \in \mathcal{H}_K^n \mid -\alpha B^v(x) + b = 0\}, \quad (5.55)$$

with $v \in \mathbb{S}^{n-1}$, $\alpha > 0$, and $b \in \mathbb{R}$. With this parameterization, the signed point-to-horosphere logit is

$$u_k(x) = \text{sign}_k \alpha_k d(x, H_{v_k, \alpha_k, b_k}), \quad (5.56)$$

Method	Logit $u_k(x)$, $\forall k \in \{1, \dots, C\}$	Space	Dist	#Params	Compact params	FLOPs	Batch efficiency
Euclidean MLR	$\langle a_k, x \rangle + b_k$, with $a_k \in \mathbb{R}^n, b_k \in \mathbb{R}$	$\mathbb{R}^n$	Real	$C(n+1)$	✓	$C(2n)$	✓
Poincaré MLR [76, Eq. (25)]	$\frac{\lambda_{p_k}^K \ a_k\ }{\sqrt{-K}} \sinh^{-1} \left(\frac{2\sqrt{-K} \langle -p_k \oplus_M x, a_k \rangle}{(1 + K \ -p_k \oplus_M x \ ^2) \ a_k\ } \right)$, with $p_k \in \mathbb{P}_K^n, a_k \in T_{p_k} \mathbb{P}_K^n$	$\mathbb{P}_K^n$	Real	$C(2n)$	✗	$C(19n+29)$	✗
Poincaré MLR [181, Eq. (6)]	$\frac{2}{\sqrt{-K}} \alpha_k \sinh^{-1}(\alpha - \beta)$, $\alpha = \lambda_x^K \sqrt{-K} \langle x, v_k \rangle \cosh(2\sqrt{-K} b_k)$, $\beta = (\lambda_x^K - 1) \sinh(2\sqrt{-K} b_k)$, with $\alpha_k > 0, v_k \in \mathbb{S}^{n-1}, b_k \in \mathbb{R}$	$\mathbb{P}_K^n$	Real	$C(n+2)$	✓	$C(4n+52)$	✓
Pseudo-Busemann MLR [162, Cor. 4.3]	$-d(x, p_k) \frac{B^{v_k}(-p_k \oplus_M x)}{\ -p_k \oplus_M x \ }$, with $p_k \in \mathbb{P}_K^n, v_k \in \mathbb{S}^{n-1}$	$\mathbb{P}_K^n$	Pseudo	$C(2n)$	✗	$C(19n+34)$	✗
Lorentz MLR [15, Eq. (12)]	$\frac{1}{\sqrt{-K}} \text{sign}(\alpha) \beta \left \sinh^{-1} \left(\sqrt{-K} \frac{\alpha}{\beta} \right) \right $, $\alpha = \cosh(\sqrt{-K} b_k) \langle z_k, x_s \rangle - \sinh(\sqrt{-K} b_k)$, $\beta = \sqrt{\cosh(\sqrt{-K} b_k) z_k \ ^2 - (\sinh(\sqrt{-K} b_k) \ z_k\)^2}$, with $z_k \in \mathbb{R}^n, b_k \in \mathbb{R}$	$\mathbb{L}_K^n$	Real	$C(n+1)$	✓	$C(4n+52)$	✓
BMLR	$-\alpha_k B^{v_k}(x) + b_k$, with $\alpha_k > 0, v_k \in \mathbb{S}^{n-1}, b_k \in \mathbb{R}$	$\mathbb{P}_K^n$ $\mathbb{L}_K^n$	Real	$C(n+2)$	✓	$\mathbb{P}_K^n : C(6n+12)$ $\mathbb{L}_K^n : C(2n+12)$	✓

Table 5.11: Comparison of C -class MLR. In Dist, **Real** means the point-to-hyperplane distance is the real distance, obtained by $\inf_{y \in H} d(x, y)$, where H is a hyperplane and d is the geodesic distance; **Pseudo** denotes a surrogate that coincides with the real distance only in Euclidean geometry. Compact params indicate whether each logit avoids an additional manifold-valued parameter. Batch efficiency indicates whether the MLR can avoid inefficient per-class loops in implementation (see Sec. A.4.2.1). In #Params, we highlight the heaviest in **red**. In FLOPs, we mark the slowest in **red** and the fastest in **green**.

where $\text{sign}_k = \text{sign}(-\alpha_k B^{v_k}(x) + b_k)$. By Thm. 133, the exact point-to-horosphere distance is

$$d(x, H_{v, \alpha, b}) = \frac{|-\alpha B^v(x) + b|}{\alpha}. \quad (5.57)$$

Consequently, Eq. (5.56) equals the BMLR logit in Eq. (5.45).

Remark 134 (Generality). Since $B^v(x) = -\langle v, x \rangle$ in Euclidean geometry, Eqs. (5.55) to (5.57) naturally generalize to their Euclidean counterparts. We also acknowledge Fan et al. [73, Eq. (2) and Prop. 3.1], who used horospheres and point-to-horosphere distances to construct a hyperbolic SVM. However, they considered only the unit Poincaré ball with curvature $K = -1$, which is a special case of Eqs. (5.55) and (5.57).

5.3.3.3 Comparison with Existing Hyperbolic MLRs

Based on the point-to-hyperplane reformulation, recent work extended MLR to the Poincaré [76, 181, 162] and Lorentz [15] models. Ganea et al. [76, Sec. 3.1] introduced the first Poincaré MLR by replacing the Euclidean point-to-hyperplane distance with its hyperbolic counterpart, where the hyperplane is defined by geodesics and the resulting

distance is the real point-to-hyperplane distance, obtained as an infimum over the hyperplane. However, the formulation is not batch efficient (see Sec. A.4.2.1). It also requires per-class parameters $a_k \in T_{p_k} \mathbb{P}_K^n$ and $p_k \in \mathbb{P}_K^n$, which leads to over-parameterization. Shimizu et al. [181, Sec. 3.1] alleviated such issues via re-parameterization. Bdeir et al. [15, Sec. 4.3] further developed a Lorentz MLR, but its hyperplanes are defined by the ambient Minkowski space, which is tailored to the Lorentz model and does not fully respect intrinsic hyperbolic geometry. Moreover, Nguyen et al. [162, Cor. 4.3] proposed a Poincaré MLR based on the Busemann function. We refer to it as Pseudo-Busemann MLR, as the induced point-to-hyperplane distance is pseudo, coinciding with the real point-to-hyperplane distance only in Euclidean geometry. It also suffers from over-parameterization and is not batch efficient.

As summarized in Tab. 5.11,⁴ BMLR unifies advantages that prior hyperbolic MLRs offer only partially. In particular, BMLR respects the real point-to-horosphere distance, uses compact parameters without an additional manifold-valued point, attains the lowest FLOPs on $\mathbb{L}_K^n$ and a competitive cost on $\mathbb{P}_K^n$, and supports batch-efficient computation. On $\mathbb{L}_K^n$, its FLOPs are even close to those of the Euclidean MLR.

5.3.4 Busemann Fully Connected Layer

We first reformulate the Euclidean FC layer by the Busemann function, then present the manifestations in the Poincaré and Lorentz models.

5.3.4.1 Formulation

As discussed in Sec. 5.2.4.2, the Euclidean FC layer can be written as

$$\bar{d}(y, H_{e_k, \mathbf{0}}) = \langle a_k, x \rangle + b_k, \quad \forall k \in \{1, \dots, m\}, \quad (5.58)$$

where $\bar{d}(y, H_{e_k, \mathbf{0}}) = \text{sign}(\langle e_k, y - \mathbf{0} \rangle) d(y, H_{e_k, \mathbf{0}})$ is the signed distance. To extend Eq. (5.58) into hyperbolic space, the right-hand side can be replaced by Eq. (5.45), as it generalizes $\langle a_k, x \rangle + b_k$. For the left-hand side, a natural idea is to use the signed point-to-horosphere distance. However, as detailed in Sec. A.4.2.2, this may fail to admit a solution for y . We therefore follow the point-to-hyperplane distance in [76, Thm. 5] for the Poincaré model and the one in [15, Eq. (44)] for the Lorentz model. Given $x \in \mathcal{H}_K^n$, the hyperbolic BFC layer $\mathcal{F} : \mathcal{H}_K^n \ni x \mapsto y \in \mathcal{H}_K^m$ is given by solving y

⁴Relative to [162, Def. 4.2, Cor. 4.3, and App. B.1.2], the Pseudo-Busemann MLR written here includes an additional sign $-$; this is intentional and matches their official implementation.

via the following m equations:

$$\bar{d}(y, H_{e_k, e}) = u_k(x), \quad \forall k \in \{1, \dots, m\}, \quad (5.59)$$

where $u_k(x) = -\alpha_k B^{v_k}(x) + b_k$ with $\{\alpha_k > 0, v_k \in \mathbb{S}^{n-1}, b_k \in \mathbb{R}\}$ as parameters. Here, $\bar{d}(y, H_{e_k, e})$ is the hyperbolic signed distance from y to the hyperplane passing through the output-space origin $e \in \mathcal{H}_K^m$. Next, we show that the above implicit definition has an explicit solution for the output y .

Theorem 135 (Poincaré BFC). $\downarrow$ Given an input $x \in \mathbb{P}_K^n$, the Poincaré BFC layer $\mathcal{F} : \mathbb{P}_K^n \rightarrow \mathbb{P}_K^m$ is given by

$$y = \frac{\omega}{1 + \sqrt{1 - K \|\omega\|^2}}, \quad \omega = \left[\frac{\sinh(\sqrt{-K} u_k(x))}{\sqrt{-K}} \right]_{k=1}^m, \quad (5.60)$$

where $u_k(x) = -\alpha_k B^{v_k}(x) + b_k$ with $\{\alpha_k > 0, v_k \in \mathbb{S}^{n-1}, b_k \in \mathbb{R}\}$ as parameters for $k = 1, \dots, m$.

Theorem 136 (Lorentz BFC). $\downarrow$ Given an input $x \in \mathbb{L}_K^n$, the Lorentz BFC layer $\mathcal{F} : \mathbb{L}_K^n \rightarrow \mathbb{L}_K^m$ is given by

$$y = \begin{bmatrix} y_t \\ y_s \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{1}{-K} + \|y_s\|^2} \\ \frac{1}{\sqrt{-K}} \sinh(\sqrt{-K} u(x)) \end{bmatrix}, \quad (5.61)$$

where $u(x) = (u_1(x), \dots, u_m(x))^\top$ with $u_k(x) = -\alpha_k B^{v_k}(x) + b_k$. Here, $\{\alpha_k > 0, v_k \in \mathbb{S}^{n-1}, b_k \in \mathbb{R}\}$ are parameters for $k = 1, \dots, m$.

Analogously to Thm. 130, our BFC layers converge to their Euclidean counterparts as $K \rightarrow 0^-$.

Theorem 137 (Limits of BFC layers). $\downarrow$ As $K \rightarrow 0^-$, the hyperbolic BFC layer $\mathcal{H}_K^n \ni x \mapsto y \in \mathcal{H}_K^m$ reduces to a Euclidean FC layer:

$$\text{Poincaré: } y_k \xrightarrow{K \rightarrow 0^-} \alpha_k \langle v_k, x \rangle + \frac{1}{2} b_k, \quad (5.62)$$

$$\text{Lorentz: } (y_s)_k \xrightarrow{K \rightarrow 0^-} \alpha_k \langle v_k, x_s \rangle + b_k. \quad (5.63)$$

5.3. Hyperbolic Busemann Neural Networks

Method	$\mathcal{F} : \mathcal{H}_K^n \ni x \mapsto y \in \mathcal{H}_K^m$	Space	Methodology	Parameters	#Params	FLOPs
Möbius [76, Eq. (27)]	$\frac{1}{\sqrt{-K}} \tanh \left(\frac{\ Wx\ }{\ x\ } \tanh^{-1} \left(\sqrt{-K} \ x\ \right) \right) \frac{Wx}{\ Wx\ }$	$\mathbb{P}_K^n$	Tangent	$W \in \mathbb{R}^{m \times n}$	mn	$2nm + 2n + 2m + 24$
Poincaré FC [181, Eq. (7)]	$y = \frac{\omega}{1 + \sqrt{1 - K \ \omega\ ^2}}, \omega_k = \frac{\sinh(\sqrt{-K} u_k(x))}{\sqrt{-K}},$ with $u_k(x)$ in Tab. 5.11	$\mathbb{P}_K^n$	Poincaré geometry	$\alpha_k > 0, v_k \in \mathbb{S}^{n-1},$ $b_k \in \mathbb{R}, k = 1, \dots, m$	$m(n+2)$	$4nm + 71m + 4$
Lorentz FC [45, Eq. (3)]	$y = \left[\sqrt{\frac{\psi(Wx, v)^2 - 1/K}{\psi(Wx, v)}}, \right.$ $\left. \psi(Wx, v) = \lambda \sigma(v^\top x + b) \frac{W\phi(x) + b}{\ W\phi(x) + b\ } \right]$	$\mathbb{L}_K^n$	Ambient Minkowski	$W \in \mathbb{R}^{m \times (n+1)},$ $v \in \mathbb{R}^{n+1}, b \in \mathbb{R}^m, b' \in \mathbb{R}, \lambda > 0$	$m(n+1) + m + (n+1) + 2$	$2nm + 8m + 2n + 10$
BFC	$y = \frac{\omega}{1 + \sqrt{1 - K \ \omega\ ^2}}, \omega = \frac{\sinh(\sqrt{-K} u(x))}{\sqrt{-K}},$ $y_s = \frac{1}{\sqrt{-K}} \sinh(\sqrt{-K} u(x)), y_t = \sqrt{\frac{1}{-K} + \ y_s\ ^2},$ with $u_k(x) = \phi(-\alpha_k B^{v_k}(x) + b_k)$	$\mathbb{P}_K^n$ $\mathbb{L}_K^n$	Busemann	$\alpha_k > 0, v_k \in \mathbb{S}^{n-1},$ $b_k \in \mathbb{R}, k = 1, \dots, m$	$m(n+2)$	$6nm + 29m + 4$ $2nm + 30m + 2$

Table 5.12: Comparison of hyperbolic FC layers. For simplicity, BFC layers do not involve the gyroaddition and assume ϕ is the identity map, which is in line with the Möbius and Lorentz FC layers.

5.3.4.2 Generalization

Following Sec. 5.2.4.2, we extend the hyperbolic BFC by inserting the activation into Eq. (5.59):

$$\bar{d}(y, H_{e_k, e}) = \phi(u_k(x)), \quad \forall k \in \{1, \dots, m\}. \quad (5.64)$$

This is reflected in Thms. 135 and 136 by replacing every $u_k(x)$ with $\phi(-\alpha_k B^{v_k}(x) + b_k)$. Moreover, inspired by the Poincaré Möbius transformation [76, Sec. 3.2], a BFC transformation could be further followed by a gyroaddition $\oplus_{\mathcal{H}} : \mathcal{H}_K^n \ni x \mapsto \mathcal{F}(x) \oplus_{\mathcal{H}} b \in \mathcal{H}_K^m$ with $b \in \mathcal{H}_K^m$ as a gyro bias. For example, the Lorentz BFC layer is generalized as

$$\mathbb{L}_K^n \ni x \mapsto y = \left[\begin{array}{c} \sqrt{\frac{1}{-K} + \|y_s\|^2} \\ \frac{1}{\sqrt{-K}} \sinh(\sqrt{-K} u(x)) \end{array} \right] \oplus_{\mathbb{L}} b \in \mathbb{L}_K^m, \quad (5.65)$$

where $u_k(x) = \phi(-\alpha_k B^{v_k}(x) + b_k)$ with parameters $\{\alpha_k > 0, v_k \in \mathbb{S}^{n-1}, b_k \in \mathbb{R}\}_{k=1}^m$ and $b \in \mathbb{L}_K^m$.

5.3.4.3 Comparison with Existing Hyperbolic FC Layers

Tab. 5.12 compares BFC with prior hyperbolic FC layers. BFC faithfully respects hyperbolic geometry, whereas the Möbius and Lorentz FC layers apply Euclidean transformations in the tangent or ambient Minkowski space, which can distort intrinsic geometry. BFC also offers flexibility across models, while Poincaré FC and Lorentz FC are tailored to their respective models. In addition, BFC uses a comparable parameterization and maintains $\mathcal{O}(nm)$ FLOPs. On $\mathbb{L}_K^n$, its FLOPs are $\mathcal{O}(2mn)$, matching the fastest layers.

Space	Method	CIFAR-10 (Num. classes: 10)			CIFAR-100 (Num. classes: 100)			Tiny-ImageNet (Num. classes: 200)			ImageNet-1k (Num. classes: 1000)		
		Acc	Fit Time	#Params	Acc	Fit Time	#Params	Acc	Fit Time	#Params	Acc	Fit Time	#Params
$\mathbb{R}^n$	MLR	95.14 $\pm$ 0.12	10.66	5.13K	77.72 $\pm$ 0.15	10.60	51.30K	65.19 $\pm$ 0.12	69.17	102.60K	71.87	2263.12	513K
$\mathbb{P}_K^n$	PMLR	95.04 $\pm$ 0.13	11.94	5.14K	77.19 $\pm$ 0.50	12.11	51.40K	64.93 $\pm$ 0.38	71.90	102.80K	71.77	2300.11	514K
	PBMLR-P	95.23 $\pm$ 0.08	21.92	10.24K	77.78 $\pm$ 0.15	76.84	102.40K	65.43 $\pm$ 0.27	336.58	204.80K	71.46	3907.12	1024K
	BMLR-P	95.32 $\pm$ 0.14	12.01	5.14K	78.10 $\pm$ 0.35	12.13	51.40K	66.16 $\pm$ 0.19	71.98	102.80K	73.36	2300.77	514K
$\mathbb{L}_K^n$	LMLR	94.98 $\pm$ 0.12	11.55	5.13K	78.03 $\pm$ 0.21	11.72	51.30K	65.63 $\pm$ 0.10	69.27	102.60K	72.46	2277.17	513K
	BMLR-L	95.25 $\pm$ 0.02	11.08	5.14K	78.07 $\pm$ 0.26	11.22	51.40K	65.99 $\pm$ 0.14	69.19	102.80K	73.24	2276.53	514K

Table 5.13: Top-1 image classification accuracy (%) of MLR methods on the ResNet-18 backbone. The best results within each hyperbolic model are **bold**. The slowest MLR and largest parameter count are shown in **red**.

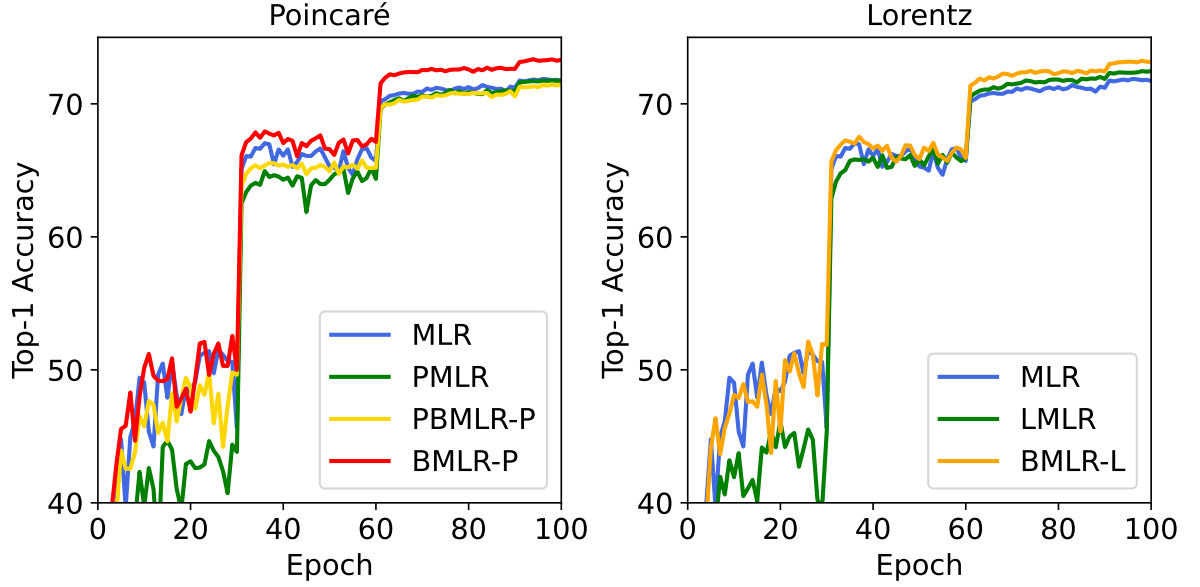


Figure 5.2: Validation accuracy curves on ImageNet-1k.

5.3.5 Experiments

We first compare BMLRs with prior hyperbolic MLRs on three architectures: ResNet-18 (image classification), CNN (genome sequences), and HGCN (node classification). We then compare BFC with prior hyperbolic FC layers on link prediction. All experiments use both the Poincaré and Lorentz models. More details on data sets and experimental settings are provided in Secs. A.1 and A.3.5.

5.3.5.1 Image Classification

Setup. Following Sec. 5.2.6.2, we use a hybrid architecture with a ResNet-18 [97] backbone and an MLR head. We compare Euclidean MLR with hyperbolic variants in both models. In Poincaré, we evaluate Poincaré MLR (PMLR) re-parameterized by Shimizu et al. [181], Pseudo-Busemann MLR (PBMLR-P) [162], and our BMLR-P. In Lorentz, we evaluate Lorentz MLR (LMLR) [15] and our BMLR-L. For hyperbolic MLRs, we

5.3. Hyperbolic Busemann Neural Networks

Benchmark	Task	Data Set	Num. classes	$\mathbb{P}_K^*$			L_K^*	
				PMLR	PBMLR-P	BMLR-P	LMLR	BMLR-L
TEB	Retrotransposons	LTR Copia	2	75.34 $\pm$ 1.02	74.37 $\pm$ 1.48	76.73 $\pm$ 1.08	73.01 $\pm$ 1.07	75.86 $\pm$ 1.52
		LINEs	2	85.54 $\pm$ 0.61	85.92 $\pm$ 0.65	86.05 $\pm$ 1.08	83.14 $\pm$ 0.80	86.72 $\pm$ 0.58
		SINEs	2	95.30 $\pm$ 0.85	95.34 $\pm$ 1.58	95.99 $\pm$ 0.74	96.70 $\pm$ 0.87	96.29 $\pm$ 0.59
	DNA transposons	CMC-EnSpm	2	83.39 $\pm$ 0.56	83.62 $\pm$ 1.00	84.03 $\pm$ 0.71	81.78 $\pm$ 1.05	84.15 $\pm$ 1.00
		hAT-Ac	2	89.38 $\pm$ 0.90	89.86 $\pm$ 0.54	89.62 $\pm$ 0.74	88.94 $\pm$ 0.69	90.70 $\pm$ 0.51
	Pseudogenes	processed	2	72.45 $\pm$ 1.49	71.99 $\pm$ 2.04	73.09 $\pm$ 1.66	73.71 $\pm$ 1.76	73.32 $\pm$ 1.65
GUE	Core Promoter Detection	tata	2	80.95 $\pm$ 1.47	79.32 $\pm$ 2.44	80.29 $\pm$ 1.63	80.90 $\pm$ 1.15	81.76 $\pm$ 1.16
		notata	2	70.02 $\pm$ 0.52	70.60 $\pm$ 0.75	70.48 $\pm$ 0.35	71.26 $\pm$ 0.56	70.43 $\pm$ 0.39
		all	2	67.64 $\pm$ 0.77	68.02 $\pm$ 0.63	68.50 $\pm$ 0.61	67.63 $\pm$ 0.56	68.36 $\pm$ 1.07
	Promoter Detection	tata	2	80.30 $\pm$ 1.59	80.27 $\pm$ 2.71	82.83 $\pm$ 1.69	83.27 $\pm$ 1.95	82.55 $\pm$ 1.54
		notata	2	92.63 $\pm$ 0.36	93.05 $\pm$ 0.32	92.75 $\pm$ 0.51	91.74 $\pm$ 0.57	92.60 $\pm$ 0.49
		all	2	90.53 $\pm$ 0.50	90.79 $\pm$ 0.77	90.20 $\pm$ 0.65	89.34 $\pm$ 0.40	89.82 $\pm$ 0.45
	Covid Variant Classification	Covid	9	74.09 $\pm$ 0.25	70.84 $\pm$ 0.80	73.40 $\pm$ 0.30	64.07 $\pm$ 0.51	72.45 $\pm$ 0.21
	Species Classification	Virus	20	67.24 $\pm$ 2.10	59.17 $\pm$ 3.32	77.12 $\pm$ 1.23	71.34 $\pm$ 2.05	77.21 $\pm$ 1.04
		Fungi	25	15.06 $\pm$ 1.32	18.75 $\pm$ 1.77	30.01 $\pm$ 0.76	15.07 $\pm$ 1.88	30.14 $\pm$ 2.48

Table 5.14: Genomic MCC of MLR methods under the CNN backbone. The best results within each hyperbolic model are **bold**.

map the ResNet-18 features to the target hyperbolic space before classification. We evaluate on CIFAR-10 [126], CIFAR-100 [126], Tiny-ImageNet [129], and ImageNet-1k [66]. On the first three data sets, we conduct five-fold experiments.

Results. Tab. 5.13 reports top-1 validation accuracy, fit time per epoch, and classifier-head parameters. Fig. 5.2 presents the ImageNet-1k accuracy curves. Overall, BMLR-P and BMLR-L consistently outperform prior hyperbolic MLRs with comparable parameters. Within each hyperbolic model, the accuracy margin over prior hyperbolic MLRs increases with the number of classes, from CIFAR-10 to CIFAR-100 and Tiny-ImageNet, with the largest gains on ImageNet-1k. This demonstrates the advantage of BMLR as task complexity increases. Besides, PBMLR-P uses approximately double the head parameters and is markedly slower due to complex batch-inefficient computation, whereas BMLR-L achieves the fastest fit time among all hyperbolic MLRs.

5.3.5.2 Genome Sequence Learning

Setup. Similar to Sec. 5.2.6.4, we evaluate hyperbolic MLRs on genome sequence learning. Following Khan et al. [119], we adopt a CNN backbone, which consists of three convolutional blocks and an MLR head. Similar to Sec. 5.3.5.1, we compare our BMLR against previous hyperbolic MLR heads by replacing the final Euclidean MLR with a hyperbolic MLR. We validate on two benchmarks: TEB [119] and *Genome Understanding Evaluation (GUE)* [234], covering a total of 16 data sets.

Results. Tab. 5.14 summarizes five-fold average MCC across TEB and GUE. Compared with other hyperbolic MLRs, our BMLR-P and BMLR-L achieve higher MCC in most tasks. Similar to Sec. 5.3.5.1, the gains are more pronounced on complex data sets

Data Set	$\mathbb{P}_K^n$			$\mathbb{L}_K^n$	
	PMLR	PBMLR-P	BMLR-P	LMLR	BMLR-L
LTR Copia	3.96	5.11	4.06	3.92	3.77
LINEs	5.80	6.90	5.73	5.50	5.32
SINEs	1.36	1.80	1.39	1.38	1.28
CMC-EnSpm	3.11	4.38	3.04	2.89	2.82
hAT-Ac	4.37	5.37	4.38	4.13	3.96
processed	5.02	5.94	4.89	4.68	4.58
unprocessed	3.30	3.90	3.29	3.05	2.95
CPD-tata	0.72	1.35	0.70	0.67	0.62
CPD-notata	6.13	12.40	6.00	5.73	5.71
CPD-all	6.85	14.35	6.59	6.35	6.33
PD-tata	0.98	1.20	0.97	1.04	0.94
PD-notata	8.33	10.53	8.29	8.11	7.84
PD-all	9.40	12.07	9.19	9.06	8.83
Covid	28.96	45.52	27.97	27.58	26.67
Virus	25.28	29.57	25.67	25.12	24.85
Fungi	6.41	8.96	6.41	6.25	6.25

Table 5.15: Fit time (s/epoch) on genome sequence learning. The fastest times are **bold** and the slowest ones are **red**.

Space	Method	Methodology	Disease $\delta = 0$	Airport $\delta = 1$	PubMed $\delta = 3.5$	Cora $\delta = 11$
$\mathbb{P}_K^n$	Möbius	Tangent	76.35 $\pm$ 1.83	93.31 $\pm$ 0.41	94.93 $\pm$ 0.06	90.80 $\pm$ 0.56
	Poincaré FC	Poincaré geometry	79.45 $\pm$ 1.01	94.31 $\pm$ 0.16	94.24 $\pm$ 0.25	88.21 $\pm$ 0.72
	BFC-P	Busemann	80.45 $\pm$ 0.93	94.88 $\pm$ 0.39	94.85 $\pm$ 0.07	91.94 $\pm$ 0.32
$\mathbb{L}_K^n$	LTFC	Tangent	71.32 $\pm$ 5.36	92.68 $\pm$ 0.35	94.85 $\pm$ 0.17	89.37 $\pm$ 0.64
	Lorentz FC	Ambient Minkowski	72.78 $\pm$ 2.04	92.99 $\pm$ 0.33	94.20 $\pm$ 0.10	92.06 $\pm$ 0.62
	BFC-L	Busemann	78.36 $\pm$ 0.51	95.37 $\pm$ 0.17	94.90 $\pm$ 0.04	92.28 $\pm$ 0.12

Table 5.16: Comparison of hyperbolic FC layers on link prediction. The best results within each hyperbolic model are **bold**.

with more classes, *e.g.*, Virus (20 classes) and Fungi (25 classes), demonstrating the effectiveness of our approach. Tab. 5.15 reports fit time per epoch, where PBMLR-P is consistently the slowest due to batch inefficiency, and BMLR-L is the fastest.

5.3.5.3 Node Classification

Setup. Following Nguyen et al. [162], we adopt the HGCN [38] backbone to evaluate our BMLR on graph data sets, including Disease [4], Airport [229], PubMed [155], and Cora [177]. The HGCN backbone consists of a hyperbolic *Graph Convolutional Network* (GCN) and an MLR as the final classification layer. Both the GCN and the MLR are built on the hyperbolic space. The vanilla HGCN uses a tangent MLR, which maps

5.3. Hyperbolic Busemann Neural Networks

Space	Method	Disease $\delta = 0$	Airport $\delta = 1$	PubMed $\delta = 3.5$	Cora $\delta = 11$
$\mathbb{P}_K^n$	HGCN	86.87 $\pm$ 2.58	85.34 $\pm$ 1.16	76.29 $\pm$ 0.98	76.56 $\pm$ 0.81
	HGCN-PMLR	88.98 $\pm$ 1.96	84.78 $\pm$ 1.48	76.02 $\pm$ 1.09	77.47 $\pm$ 1.15
	HGCN-PBMLR-P	89.05 $\pm$ 0.78	85.04 $\pm$ 0.97	75.89 $\pm$ 0.78	77.90 $\pm$ 1.00
	HGCN-BMLR-P	92.45 $\pm$ 0.96	86.02 $\pm$ 0.53	77.36 $\pm$ 0.73	78.48 $\pm$ 1.52
$\mathbb{L}_K^n$	HGCN	87.83 $\pm$ 0.77	84.94 $\pm$ 1.40	76.49 $\pm$ 0.88	77.37 $\pm$ 1.72
	HGCN-LMLR	89.72 $\pm$ 1.51	82.61 $\pm$ 1.01	75.44 $\pm$ 1.17	69.91 $\pm$ 3.61
	HGCN-BMLR-L	90.80 $\pm$ 1.15	85.27 $\pm$ 1.17	77.30 $\pm$ 0.41	77.65 $\pm$ 2.10

Table 5.17: Node classification F1 scores of hyperbolic MLRs on the HGCN backbone, where δ denotes graph hyperbolicity (lower is more hyperbolic). The best results within each hyperbolic model are **bold**.

features into the tangent space via Log_e and applies a Euclidean MLR. We replace this with different hyperbolic MLRs.

Results. Tab. 5.17 reports average F1 scores. Our BMLRs consistently outperform prior hyperbolic MLRs within each hyperbolic model. As graphs become less hyperbolic, that is, for larger δ , existing hyperbolic heads could underperform the vanilla tangent-based MLR, for example, PBMLR-P on PubMed, and LMLR on Airport, PubMed, and Cora. Especially on Cora, which has the largest δ , LMLR lags the tangent baseline by a large margin (69.91 vs. 77.37). In contrast, BMLR remains the top performer across all δ values, indicating that Busemann-based decoding robustly strengthens HGCN over a broader range of graph hyperbolicity.

5.3.5.4 Link Prediction

Setup. We compare our BFC layers with prior hyperbolic FC layers, including the Möbius layer [76] that operates via the tangent space, the Lorentz FC layer [45] that operates through the ambient Minkowski space, and the Poincaré FC layer [181]. Mimicking the Möbius layer, we also implement a Lorentz tangent FC layer, $\text{Exp}_{\mathbf{0}}(M \text{Log}_{\mathbf{0}}(x))$, referred to as LTFC. Following Chami et al. [38], we evaluate on Disease, Airport, PubMed, and Cora. Following the HNN implementation [76, 38], all methods share the same backbone with two FC layers. For a fair comparison, all hyperbolic FC layers are followed by a gyroaddition biasing. For BFC, we use $\phi = \tanh$ on Airport and Cora and the identity map on the other two data sets.

Results. Tab. 5.16 reports five-fold test AUC. Our BFC layers generally outperform prior hyperbolic FC layers. The gains are most pronounced on Disease, which is the most hyperbolic ($\delta = 0$), where Busemann-based decoding is markedly more effective than tangent or ambient methods, indicating better capture of intrinsic hyperbolic

Space	Method	Disease		Airport		PubMed		Cora	
		Fit Time	#Params	Fit Time	#Params	Fit Time	#Params	Fit Time	#Params
$\mathbb{P}_K^n$	Möbius	0.0200	464	0.0535	480	0.1120	8288	0.0229	23216
	Poincaré FC	0.0198	528	0.0536	544	0.1176	8352	0.0248	23280
	BFC-P	0.0201	528	0.0512	544	0.1123	8352	0.0231	23280
$\mathbb{L}_K^n$	LTFC	0.0343	464	0.0818	480	0.1633	8288	0.0370	23216
	Lorentz FC	0.0232	563	0.0715	580	0.1537	8876	0.0261	24737
	BFC-L	0.0244	528	0.0713	544	0.1525	8352	0.0280	23280

Table 5.18: Efficiency comparison: fit time (s/epoch) and parameter count. Slowest results and largest parameter counts are in **red**.

geometry. This observation aligns with geometric intuition, since tangent space or ambient space approximations inherently struggle to represent curved manifolds in highly non-Euclidean cases.

Training Time and Parameter Count. Tab. 5.18 summarizes fit time per epoch and parameter counts. Our BFC layers achieve training time and model size comparable to existing layers. In particular, LTFC is the slowest due to costly logarithmic and exponential maps, and LFC uses the largest number of parameters among Lorentz variants.

5.4 Full-Rank Correlation Networks

5.4.1 Introduction

The preceding two sections studied manifold-specific designs for hyperbolic learning. We now turn to neural networks on full-rank correlation manifolds. Covariance matrices in the SPD manifold have achieved success in various applications, with many deep network architectures adapted to leverage their Riemannian geometries [106, 31, 37, 59, 167, 123, 206, 47, 118, 134, 172, 215, 115, 104]. In contrast, correlation matrices, despite serving as statistically compact alternatives to covariance matrices [7], remain unexplored in deep learning.

As discussed in Sec. 2.9.2, Riemannian structures for correlation matrices have only recently been developed. David and Gu [62] identified full-rank correlation matrices as a quotient manifold of the SPD manifold, referred to as the correlation manifold. However, this quotient geometry does not guarantee uniqueness or closed forms of the Riemannian logarithm and Fréchet mean [195, Sec. 1.1]. To close this gap, Thanwerdas and Pennec [195] proposed three theoretically and computationally convenient geometries: ECM, LECM, and PHCM. Thanwerdas [191] further introduced two efficient permutation-invariant metrics: OLM and LSM. These Riemannian structures provide

promising foundations for extending Euclidean deep learning to the correlation manifold.

On the other hand, several fundamental layers in Euclidean deep learning, such as MLR, FC, and convolutional layers, have been extended to different manifolds by leveraging their rich Riemannian or algebraic structures [106, 107, 108, 76, 37, 45, 181, 15, 51, 161]. For the SPD manifold, these layers have been constructed using bilinear mapping [106], weighted Fréchet means [37], gyrovectors spaces [159, 161], and Riemannian geometry [49, 51].

Inspired by these advancements, we develop MLR, FC, and convolutional layers for correlation manifolds in a geometrically intrinsic manner. We begin by systematically introducing four types of correlation-based MLR, FC, and convolutional layers, corresponding to ECM, LECM, OLM, and LSM, respectively. Besides, we discuss backpropagation through Riemannian computations over the correlation manifold, with novel approaches for accurate backpropagation under OLM and LSM. As the above four metrics have zero curvature, our next focus is to build correlation layers under the geometry of non-zero curvature. We target PHCM, induced by the product of multiple hyperbolic spaces [195, Thm. 4.4]. By adapting existing Poincaré-based hyperbolic MLR, FC, and convolutional layers designed for a single Poincaré ball [76, 181], we construct their counterparts on the correlation manifold. Together with the corresponding backpropagation mechanisms, these layers constitute complete *Correlation Networks* (*CorNets*) under different geometries. The effectiveness is validated by experiments comparing our approach against existing SPD and Grassmannian baselines.

Tab. 5.19 summarizes the correspondence between Euclidean and our correlation layers. In summary, our **main contributions** are as follows:

- (1) We systematically extend MLR, FC, and convolutional layers to the correlation manifold under five geometries: four with zero curvature and one with non-zero curvature. The developed layers enable flexible variation of the latent geometry under a consistent network architecture, allowing for straightforward comparisons across different correlation geometries.
- (2) We develop accurate backpropagation of Riemannian computations under OLM and LSM.
- (3) We conduct experiments against existing SPD and Grassmannian networks to demonstrate the effectiveness of correlation embeddings and networks.

Outline. Sec. 5.4.2 constructs correlation MLR, FC, and convolutional layers under four flat geometries. Sec. 5.4.3 develops their counterparts under a non-zero-curvature

Space	Euclidean $\mathbb{R}^n$	Correlation $\text{Cor}^+(n)$
C -class MLR	$f : \mathbb{R}^n \ni x \mapsto p = \text{softmax}(Ax + b) \in \mathbb{R}^C$	$f : \text{Cor}^+(n) \ni X \mapsto p \in \mathbb{R}^C$
FC layer	$\mathcal{F} : \mathbb{R}^n \ni x \mapsto y = Ax + b \in \mathbb{R}^m$	$\mathcal{F} : \text{Cor}^+(n) \ni X \mapsto Y \in \text{Cor}^+(m)$
Convolution	Kernel-based FC in each receptive field	Kernel-based correlation FC in each receptive field
Geometry	Euclidean	ECM, LECM, OLM, LSM and PHCM

Table 5.19: Correspondence between Euclidean and correlation-based layers. For convolution, kernel-based FC refers to applying a convolution kernel to a receptive field, which is an FC transformation.

geometry and establishes the order-invariance of the associated β -operations. Sec. 5.4.4 presents backpropagation over the correlation geometries, and Sec. 5.4.5 evaluates Cor-Nets under these five geometries. Proofs are deferred to Sec. B.8.

5.4.2 Log-Euclidean Correlation Layers

Since ECM, LECM, OLM, and LSM are derived via diffeomorphisms from Euclidean spaces, they are collectively termed Log-Euclidean metrics [191]. This motivates the principled development of MLR, FC, and convolutional layers [191].

5.4.2.1 Log-Euclidean Correlation MLRs

As discussed in Sec. 4.3.2.1, the MLR can be rewritten by point-to-hyperplane formulations. We use the same margin-distance infimum, while the Euclidean isometries of ECM, LECM, OLM, and LSM make it possible to solve this infimum exactly under all four geometries. To avoid over-parameterization, we follow Sec. 5.2.4.1 and set $P_k = \text{Exp}_E(\gamma_k[Z_k])$ and $A_k = \text{PT}_{E \rightarrow P_k}(Z_k)$, with $[Z_k] = \frac{Z_k}{\|Z_k\|_E}$ as the unit direction vector of Z_k . Here, E is the origin of $\mathcal{M}$, while $\gamma_k \in \mathbb{R}$ and $Z_k \in T_E \mathcal{M} \cong \mathbb{R}^m$ are the MLR parameters. This is a concrete instance of the trivialization strategy reviewed in Sec. 2.7. Under this trivialization, each hyperplane H_{A_k, P_k} is denoted as H_{Z_k, γ_k} .

As all Log-Euclidean metrics are isometric to Euclidean spaces, the corresponding MLRs admit principled closed forms.

Theorem 138. [↓] *Let $(\mathcal{M}, g^{\mathcal{M}})$ be an m -dimensional manifold that is isometric to the standard Euclidean space $\mathbb{R}^m$ via the diffeomorphism $\phi : \mathcal{M} \rightarrow \mathbb{R}^m$. Denoting $E = \phi^{-1}(\mathbf{0})$ with $\mathbf{0}$ as the zero vector, each $v_k(X)$ and margin hyperplane H_{Z_k, γ_k} in the C -class Riemannian MLR are $v_k(X) = \langle \phi(X), \phi_{*,E}(Z_k) \rangle - \gamma_k \|\phi_{*,E}(Z_k)\|$ and $H_{Z_k, \gamma_k} = \{X \in \mathcal{M} \mid v_k(X) = 0\}$, respectively. Here, $Z_k \in T_E \mathcal{M} \cong \mathbb{R}^m$ and $\gamma_k \in \mathbb{R}$ for $1 \leq k \leq C$ are MLR parameters, while ϕ_* is the differential.*

Simple computations show that

$$\begin{aligned} \text{ECM: } \phi^{\text{EC}}(I) &= \mathbf{0}, & \text{LECM: } \log \circ \Theta(I) &= \mathbf{0}, \\ \text{OLM: } \text{Log}^\circ(I) &= \mathbf{0}, & \text{LSM: } \text{Log}^\star(I) &= \mathbf{0}. \end{aligned} \quad (5.66)$$

Therefore, we define the origin of the correlation manifold under four Log-Euclidean metrics as the identity matrix. Besides, Thm. 138 suggests that Log-Euclidean MLRs can be obtained modulo the calculation of diffeomorphisms and their differentials at the identity matrix I .

Proposition 139 (Differentials). $\Downarrow$ For any tangent vector $V \in T_I \text{Cor}^+(n) \cong \text{Hol}(n)$, the differentials of ϕ^{EC} , $\log \circ \Theta$, Log° , and $\text{Log}^\star$ at the identity matrix I are

$$\begin{aligned} \phi_{*,I}^{\text{EC}}(V) &= \lfloor V \rfloor, & (\log \circ \Theta)_{*,I}(V) &= \lfloor V \rfloor, \\ \text{Log}_{*,I}^\circ(V) &= V, & \text{Log}_{*,I}^\star(V) &= V - \text{diag}(V\mathbf{1}), \end{aligned} \quad (5.67)$$

where $\text{diag} : \mathbb{R}^n \rightarrow \text{Diag}(n)$ returns a diagonal matrix, and $\mathbf{1} = (1, \dots, 1)^\top \in \mathbb{R}^n$.

Putting Thm. 139 into Thm. 138, we obtain correlation MLRs under four Log-Euclidean metrics.

Theorem 140 (Log-Euclidean MLRs). Given $C \in \text{Cor}^+(n)$, the logits $v_k(C)$ for the k -th class in the correlation MLRs under four Log-Euclidean metrics are

$$\begin{aligned} v_k^{\text{EC}}(C) &= \langle \lfloor \Theta(C) \rfloor, \lfloor Z_k \rfloor \rangle - \gamma_k \|\lfloor Z_k \rfloor\|, \\ v_k^{\text{LEC}}(C) &= \langle \log \circ \Theta(C), \lfloor Z_k \rfloor \rangle - \gamma_k \|\lfloor Z_k \rfloor\|, \\ v_k^{\text{OL}}(C) &= \langle \text{Log}^\circ(C), Z_k \rangle - \gamma_k \|Z_k\|, \\ v_k^{\text{LS}}(C) &= \langle \text{Log}^\star(C), \text{Log}_{*,I}^\star(Z_k) \rangle - \gamma_k \|\text{Log}_{*,I}^\star(Z_k)\|, \end{aligned} \quad (5.68)$$

where $Z_k \in \text{Hol}(n)$ and $\gamma_k \in \mathbb{R}$ are parameters.

5.4.2.2 Log-Euclidean FC and Convolutional Layers

Following Sec. 5.2.4.2, we now generalize this point-to-hyperplane FC construction to the correlation manifold.

Definition 141 (Correlation FC layers). Given a metric g , the correlation FC layer $\mathcal{F} : \text{Cor}^+(n) \ni X \mapsto Y \in \text{Cor}^+(m)$ returns the output Y by solving the following

$d = m(m-1)/2$ equations:

$$s_k d(Y, H_{O_k, I}) = v_k(X; Z_k, \gamma_k), \quad 1 \leq k \leq d, \quad (5.69)$$

where $s_k = \text{sign}(\langle \text{Log}_I(Y), O_k \rangle_I)$, I is the identity matrix, d is the dimension of $\text{Cor}^+(m)$, $\{O_k\}_{k=1}^d$ is an orthonormal basis over $T_I \text{Cor}^+(m)$, $d(\cdot, \cdot)$ is the margin distance to the hyperplane $H_{O_k, I}$, and v_k is defined by Sec. 4.3.2.1 for $\text{Cor}^+(n)$. The FC parameters are $\{Z_k \in \text{Hol}(n)\}_{k=1}^d$ and $\{\gamma_k \in \mathbb{R}\}_{k=1}^d$.

Sec. A.4.3.1 details how Thm. 141 extends the existing SPD, Poincaré, and Euclidean FC layers. Although Thm. 141 is implicitly defined by d equations, the FC layers under four Log-Euclidean geometries admit explicit expressions in a principled manner. Analogous to Thm. 138, a corresponding result for the FC layer is presented in Thm. 201, which yields the Log-Euclidean FC layers.

Theorem 142 (Log-Euclidean FC layers). $\downarrow$ Given an input correlation $C \in \text{Cor}^+(n)$, the correlation FC layers $\mathcal{F}(\cdot) : \text{Cor}^+(n) \rightarrow \text{Cor}^+(m)$ under different Log-Euclidean metrics are

$$ECM: Y = \text{Cor} \circ \text{Chol}^{-1} (V^{\text{EC}} + I_m), \quad (5.70)$$

$$LECM: Y = \text{Cor} \circ \text{Chol}^{-1} \circ \exp (V^{\text{LEC}}), \quad (5.71)$$

$$OLM: Y = \text{Exp}^\circ (V^{\text{OL}}), \quad (5.72)$$

$$LSM: Y = \text{Cor} \circ \exp (V^{\text{LS}}), \quad (5.73)$$

where the (i, j) -th elements in $V^{\text{EC}} \in \text{LT}^0(m)$, $V^{\text{LEC}} \in \text{LT}^0(m)$, $V^{\text{OL}} \in \text{Hol}(m)$, and $V^{\text{LS}} \in \text{Row}_0(m)$ are

$$V_{ij}^{\text{EC}} = \begin{cases} v_{ij}^{\text{EC}}(C), & \text{if } i > j \\ 0, & \text{otherwise} \end{cases} \quad (5.74)$$

$$V_{ij}^{\text{LEC}} = \begin{cases} v_{ij}^{\text{LEC}}(C), & \text{if } i > j \\ 0, & \text{otherwise} \end{cases} \quad (5.75)$$

$$V_{ij}^{\text{OL}} = \begin{cases} \frac{v_{ij}^{\text{OL}}(C)}{\sqrt{2}}, & \text{if } i > j \\ V_{ji}^{\text{OL}}, & \text{if } i < j \\ 0, & \text{otherwise} \end{cases} \quad (5.76)$$

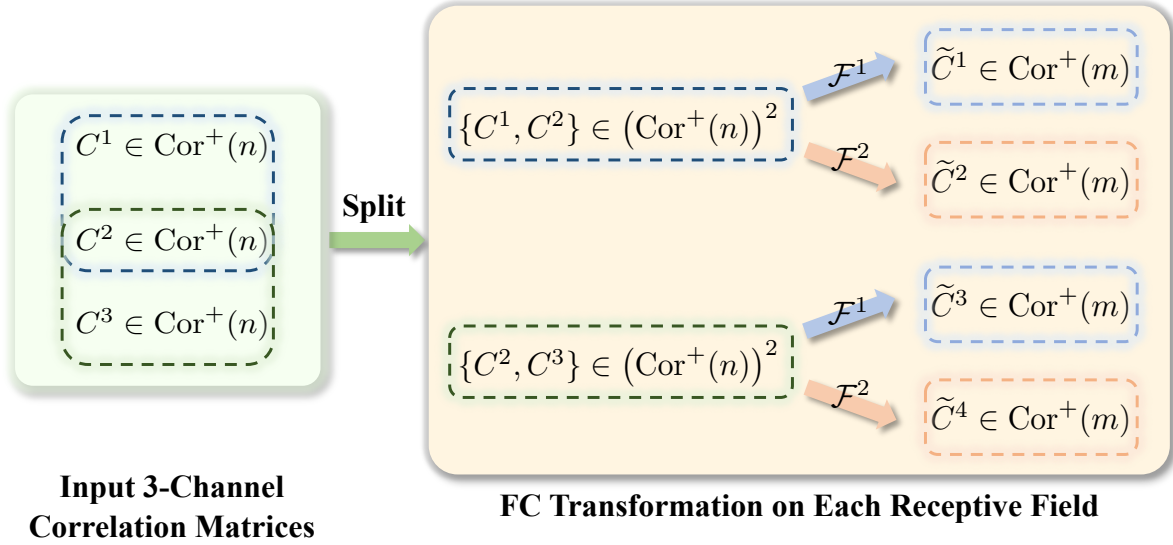


Figure 5.3: Illustration of the Log-Euclidean 1D convolution with two kernels. The 3-channel input is first split into two receptive fields along the channel dimension. In each receptive field, two kernels are applied to the product space.

$$V_{ij}^{\text{LS}} = \begin{cases} v_{ij}^{\text{LS}}(C)/\sqrt{6}, & \text{if } m > i > j \geq 1 \\ v_{ii}^{\text{LS}}(C)/\sqrt{3}, & \text{if } m > i \geq 1 \\ V_{ji}^{\text{LS}}, & \text{if } i < j \\ -\sum_{k=1}^{m-1} V_{kj}^{\text{LS}}, & \text{if } i = m, 1 \leq j < m \\ \sum_{k=1}^{m-1} \sum_{l=1}^{m-1} V_{lk}^{\text{LS}}, & \text{if } i = j = m \end{cases} \quad (5.77)$$

Each v_{ij}^g with $g \in \{\text{EC}, \text{LEC}, \text{OL}, \text{LS}\}$ is defined by Eq. (5.68) with parameters $Z_{ij} \in \text{Hol}(n)$ and $\gamma_{ij} \in \mathbb{R}$. For v_{ij}^{EC} , v_{ij}^{LEC} , and v_{ij}^{OL} , the indices satisfy $i, j = 1, \dots, m$ and $i > j$. For v_{ij}^{LS} , they satisfy $i, j = 1, \dots, m-1$ and $i \geq j$.

Correlation Convolution. Following the convolution-as-FC construction in Sec. 5.2.4.3, we develop the correlation convolution. The c -channel correlation matrices $\{C_i \in \text{Cor}^+(n)\}_{i=1}^c$ within a receptive field are first concatenated into $\mathbf{C} \in (\text{Cor}^+(n))^c$. For each convolution kernel, $\mathbf{C}$ is then fed into a correlation FC layer.⁵ Fig. 5.3 illustrates the above process.

⁵Thm. 142 naturally supports product geometries, which are detailed in Sec. A.4.3.2.

5.4.3 Poly-Hyperbolic-Cholesky Layers

As detailed in Sec. 2.9.2, the space $\mathcal{L}^n$, consisting of the Cholesky factors of $\text{Cor}^+(n)$, can be identified with the product of $n - 1$ hyperbolic open hemispheres, $\mathbb{PHS}^{n-1} = \prod_{i=1}^{n-1} \text{HS}^i$. We focus on the widely used hyperbolic Poincaré ball, whose MLR, FC, and β -concatenation components are reviewed in Sec. A.2.4. In the following, we focus on the canonical Poincaré ball ($K = -1$), namely the unit Poincaré ball $\mathbb{P}^n$. We first identify the correlation manifold with the poly-Poincaré space $\mathbb{PP}^{n-1} = \prod_{i=1}^{n-1} \mathbb{P}^i$, the product of $n - 1$ unit Poincaré balls. Then, we develop correlation layers from the layers on a single Poincaré space.

5.4.3.1 Correlation Geometry via Poincaré Balls

Proposition 143 (Isometries). [↓] *The open hemisphere HS^n is isometric to the unit Poincaré ball $\mathbb{P}^n$ by*

$$\begin{aligned} \psi_{\text{HS}^n \rightarrow \mathbb{P}^n}((x^\top, x_{n+1})^\top) &= \frac{x}{1 + x_{n+1}}, \\ \psi_{\mathbb{P}^n \rightarrow \text{HS}^n}(y) &= \frac{1}{1 + \|y\|^2} \begin{pmatrix} 2y \\ 1 - \|y\|^2 \end{pmatrix}, \end{aligned} \quad (5.78)$$

with $(x^\top, x_{n+1})^\top \in \text{HS}^n \subset \mathbb{R}^n \times \mathbb{R}^+$ and $y \in \mathbb{P}^n \subset \mathbb{R}^n$.

Thm. 143 indicates that $\text{Cor}^+(n)$ can be identified with $\mathbb{PP}^{n-1} = \prod_{i=1}^{n-1} \mathbb{P}^i$ via the diffeomorphism Φ :

$$C \xrightarrow{\text{Chol}} \begin{pmatrix} 1 & 0 & \cdots & 0 \\ L_{21} & L_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n1} & L_{n2} & \cdots & L_{nn} \end{pmatrix} \xrightarrow{\prod_{i=1}^{n-1} \psi_i} \begin{pmatrix} \psi_1(h_1) \\ \vdots \\ \psi_{n-1}(h_{n-1}) \end{pmatrix} \quad (5.79)$$

with $C \in \text{Cor}^+(n)$, $h_i = (L_{i+1,1}, \dots, L_{i+1,i+1})^\top \in \text{HS}^i$, and $\psi_i = \psi_{\text{HS}^i \rightarrow \mathbb{P}^i}$. This identification motivates us to construct the correlation layers using the corresponding layers over Poincaré spaces.

5.4.3.2 Revisiting Poincaré Layers

The Poincaré MLR and FC layers are reviewed in Sec. A.2.4 and follow the point-to-hyperplane logic discussed in Chapter 4 and Secs. 5.2 and 5.3. The convolutional

5.4. Full-Rank Correlation Networks

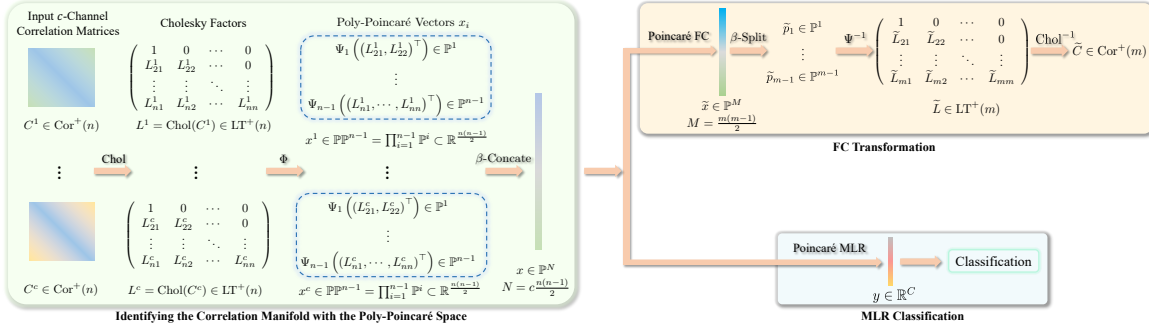


Figure 5.4: Illustration of the PHCM convolution and MLR. The multi-channel input correlation matrices are denoted as $\{C^i\}_{i=1}^c$. For the convolutional layer, the illustration focuses on the transformation within a receptive field and assumes a single-channel output.

construction uses the Poincaré β -concatenation defined below.

The Poincaré convolutional layer shares a logic similar to the correlation convolution, except it uses β -concatenation to concatenate the Poincaré vectors in each receptive field [181, Secs. 3.3–3.4], which can stabilize the norm of the Poincaré vector. The Poincaré β -concatenation generalizes the Euclidean concatenation via the scaled concatenation in the tangent space. Given inputs $\{x_i \in \mathbb{P}^{n_i}\}_{i=1}^N$, it is defined as $\text{Exp}_0(\beta_n(\beta_{n_1}^{-1}v_1^\top, \dots, \beta_{n_N}^{-1}v_N^\top))^\top \in \mathbb{P}^n$, where $v_i = \text{Log}_0(x_i)$ and $n = \sum_{i=1}^N n_i$. Here, β_{n_i} and β_n are defined by the beta function $\beta_\alpha = B(\alpha/2, 1/2)$. The inverse is called the Poincaré β -split. The Poincaré convolution is: (1) β -concatenating the multi-channel feature in a given receptive field; and (2) performing the Poincaré FC transformation.

5.4.3.3 Building Poly-Hyperbolic-Cholesky Layers

PHCM MLR. The input multi-channel correlation matrices, $\mathbf{C} = \{C^i \in \text{Cor}^+(n)\}_{i=1}^c$, are first mapped into poly-Poincaré spaces as $\mathbf{x} = \{x^i = \Phi(C^i) \in \mathbb{P}^{n-1}\}_{i=1}^c$. The resulting Poincaré vectors are then β -concatenated into a single Poincaré vector $x \in \mathbb{P}^N$, where $N = c \frac{n(n-1)}{2}$. This concatenated vector is subsequently fed into the Poincaré MLR for classification.

PHCM Convolutional and FC Layer. The convolutional layer follows a logic similar to Log-Euclidean convolution. The multi-channel correlation matrices within a receptive field $\mathbf{C} = \{C^i \in \text{Cor}^+(n)\}_{i=1}^c$ are first mapped to a β -concatenated Poincaré vector $x \in \mathbb{P}^N$ as in the PHCM MLR, which is then fed into the Poincaré FC layer for dimensionality transformation. This produces a vector $\tilde{x} \in \mathbb{P}^M$, with $M = k \frac{m(m-1)}{2}$, which is then split using β -split. Subsequently, applying Φ^{-1} reconstructs new $k \times m \times m$

correlation matrices. When the input is a single correlation matrix, it is reduced to the correlation FC.

Fig. 5.4 illustrates the PHCM layers. However, there is an underlying ambiguity in the above discussion. To clarify, we write each $x^i \in \mathbb{P}\mathbb{P}^{n-1}$ in $\mathbf{x}$ as $x^i = \{p_1^i \in \mathbb{P}^1, \dots, p_{n-1}^i \in \mathbb{P}^{n-1}\}$, which gives $\mathbf{x} = \{p_j^i \in \mathbb{P}^j\}_{i=1, j=1}^{i=c, j=n-1}$. We can either concatenate twice by $i \rightarrow j$ or once along both i and j . A similar issue arises with β -split. The following theorem establishes this invariance.

Theorem 144 (Order-invariance). $\llbracket \downarrow \rrbracket$ *Given multichannel data $x_{i_1, \dots, i_n} \in \mathbb{P}^{n_{i_n}}$ with $i_j \in \{1, \dots, N_j\}$, applying the β -concatenation sequentially n times in the order $i_n \rightarrow \dots \rightarrow i_1$ is equivalent to a single β -concatenation along all indices simultaneously. Similarly, β -splitting $x \in \mathbb{P}^N$ into multichannel data $x_{i_1, \dots, i_n} \in \mathbb{P}^{n_{i_n}}$ with $i_j \in \{1, \dots, N_j\}$ and $N = \left(\prod_{j=1}^{n-1} N_j\right) \sum_{i_n=1}^{N_n} n_{i_n}$ under the sequential order $i_1 \rightarrow \dots \rightarrow i_n$ is identical to the one under a single β -split to generate all indices simultaneously.*

Therefore, we always conduct the β -operation simultaneously along both i and j .

5.4.4 Backpropagation over Correlation Geometries

Except for $\mathcal{D}$ and $\mathcal{D}^*$, all computations involved in the five metrics can be backpropagated using existing techniques or PyTorch's auto-differentiation. The matrix logarithm, matrix exponentiation, and Cholesky decomposition, together with their differentials and backpropagation, are reviewed in Sec. 2.8. It therefore remains to discuss $\mathcal{D}$ and $\mathcal{D}^*$.

$\mathcal{D}$ and $\mathcal{D}^*$. Their gradients can be backpropagated either approximately through their iterative algorithms or accurately using the following two propositions.

Proposition 145 (Gradients w.r.t. $\mathcal{D}$). $\llbracket \downarrow \rrbracket$ *Let $l(\cdot)$ be the loss function and define $F : \text{Hol}(n) \rightarrow \mathcal{S}^n$ by $F(H) = Y = \mathcal{D}(H) + H$ for any symmetric hollow matrix H , where $\mathcal{S}^n$ is the Euclidean space of $n \times n$ symmetric matrices. Let $Y = U \Delta U^\top$ be the eigendecomposition with $(\delta_1, \dots, \delta_n)$ as eigenvalues. Given the succeeding gradient $\frac{\partial l}{\partial Y}$, the output gradient $\frac{\partial l}{\partial H}$ is*

$$\frac{\partial l}{\partial H} = \text{off} \left(\frac{\partial l}{\partial Y} - \exp_{*,Y} \left(\mathbb{D} \left((H^0)^{-1} \text{Dv} \left(\frac{\partial l}{\partial Y} \right) \mathbf{1}^\top \right) \right) \right), \quad (5.80)$$

with $H^0 \in \mathcal{S}_{++}^n$ having entries $H_{il}^0 = \sum_{j,k} U_{ij} U_{ik} U_{lj} U_{lk} [L_{\text{exp}}]_{j,k}$, where L_{exp} is the

Loewner matrix in Eq. (2.91) specialized to $f = \exp$ and $\sigma_i = \delta_i$. Here, $\mathbb{D}(\cdot) : \mathbb{R}^{n \times n} \rightarrow \text{Diag}(n)$ extracts the diagonal matrix, while $\text{Dv}(\cdot) : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^n$ returns a vector of diagonal elements. Besides, $\text{off}(\cdot)$ subtracts the diagonal matrix from a matrix, and $\exp_{*,Y}$ is the differential of the symmetric matrix exponential given by Eq. (2.90).

Proposition 146 (Gradients w.r.t. $\mathcal{D}^*$). [↓] Following the notation in Thm. 145, define $F : \text{Cor}^+(n) \rightarrow \text{Row}_1^+(n)$ by $F(C) = \Sigma = \mathcal{D}^*(C)C\mathcal{D}^*(C)$, where $\text{Row}_1^+(n)$ is the manifold of $n \times n$ SPD matrices with unit row sum. Given the succeeding gradient $\frac{\partial l}{\partial \Sigma}$, the output gradient $\frac{\partial l}{\partial C}$ is

$$\frac{\partial l}{\partial C} = \Delta \left(\frac{\partial l}{\partial \Sigma} - ((I + \Sigma)^{-1} \tilde{v} \mathbf{1}^\top)_{\text{sym}} \right) \Delta, \quad (5.81)$$

where $\Delta = \mathbb{D}(\Sigma)^{1/2}$, $\tilde{v} = \text{Dv}(\Sigma \frac{\partial l}{\partial \Sigma} + \frac{\partial l}{\partial \Sigma} \Sigma)$, I is the identity matrix, and $\mathbf{1} \in \mathbb{R}^n$ is the vector with all entries equal to 1. Here, $(A)_{\text{sym}} = \frac{A + A^\top}{2}$.

5.4.5 Experiments

We construct Riemannian networks on the correlation manifold, termed CorNets, using the proposed convolutional and MLR layers. Following previous work [106, 31, 50], we evaluate our approach on the Radar data set [31] for radar signal classification, along with the HDM05 [153], FPHA [80] and NTU120 [138] data sets for human action recognition. More details on data sets and experimental settings are provided in Secs. A.1 and A.3.6.

Implementation. We denote CorNet-Metric as the CorNet composed of correlation convolution and MLR layers under a specified metric. In line with Nguyen et al. [161], each CorNet consists of one correlation convolutional layer followed by a correlation MLR layer, trained with cross-entropy loss. Following Wang et al. [210], Nguyen et al. [161], each raw feature is modeled as a multi-channel $[c, n, n]$ SPD tensor. Since matrix power effectively activates SPD matrices by deforming their geometry, as detailed in Sec. 4.3.3.1 and prior work [194, 53], we first apply a matrix power, and then convert the result to correlation matrices as the input of CorNet. Due to trivialization, all trainable manifold-valued parameters are represented by Euclidean parameters and optimized by standard Euclidean optimizers. We compare CorNets against representative Grassmannian and SPD networks, including GrNet [108], GyroGr [159],

Manifold	Method	Radar		HDM05		FPFA		NTU120	
		Mean $\pm$ STD	Time	Mean $\pm$ STD	Time	Mean $\pm$ STD	Time	Mean $\pm$ STD	Time
Grassmann	GrNet [108]	90.48 $\pm$ 0.76	1.39	63.19 $\pm$ 0.70	1.64	85.31 $\pm$ 0.90	0.70	57.59 $\pm$ 0.22	50.97
	GyroGr* [159]	90.64 $\pm$ 0.57	1.38	58.32 $\pm$ 1.23	2.48	79.62 $\pm$ 0.49	0.70	53.76 $\pm$ 0.18	136.96
	GyroGr-Scaling* [159]	88.88 $\pm$ 1.52	1.63	39.75 $\pm$ 0.93	3.52	58.62 $\pm$ 1.66	1.03	43.90 $\pm$ 0.23	338.01
SPD	SPDNet [106]	93.25 $\pm$ 1.10	0.66	64.57 $\pm$ 0.61	0.50	85.59 $\pm$ 0.72	0.28	51.25 $\pm$ 0.36	12.77
	SPDNetBN [31]	94.85 $\pm$ 0.99	1.25	71.28 $\pm$ 0.79	0.94	89.33 $\pm$ 0.49	0.58	54.35 $\pm$ 0.43	19.78
	SPDResNet-AIM [118]	95.71 $\pm$ 0.37	0.96	64.95 $\pm$ 0.82	1.23	86.63 $\pm$ 0.55	0.69	57.33 $\pm$ 0.35	23.84
	SPDResNet-LEM [118]	95.89 $\pm$ 0.86	0.77	70.12 $\pm$ 2.45	0.55	85.07 $\pm$ 0.99	0.30	61.34 $\pm$ 2.02	13.00
	SPDNetLieBN-AIM [50]	95.47 $\pm$ 0.90	1.21	71.83 $\pm$ 0.69	1.15	90.39 $\pm$ 0.66	0.97	58.20 $\pm$ 0.46	31.10
	SPDNetLieBN-LCM [50]	94.80 $\pm$ 0.71	1.10	71.78 $\pm$ 0.44	1.11	86.33 $\pm$ 0.43	0.59	57.96 $\pm$ 0.43	22.06
	SPDNetMLR [51]	94.59 $\pm$ 0.82	0.66	65.90 $\pm$ 0.93	5.46	85.60 $\pm$ 0.43	0.88	58.59 $\pm$ 0.13	22.48
	GyroLE* [159]	96.24 $\pm$ 0.24	0.79	73.17 $\pm$ 0.37	2.86	90.73 $\pm$ 0.92	1.59	59.29 $\pm$ 0.42	22.08
	GyroLC* [159]	93.60 $\pm$ 1.31	0.66	67.53 $\pm$ 0.85	1.49	76.10 $\pm$ 0.63	0.78	59.29 $\pm$ 0.42	14.14
	GyroAI* [159]	96.29 $\pm$ 0.48	0.99	72.34 $\pm$ 1.06	22.80	89.60 $\pm$ 0.37	12.62	62.21 $\pm$ 0.29	98.31
	GyroSPD++* [161]	95.20 $\pm$ 0.88	5.09	69.82 $\pm$ 1.79	103.57	89.50 $\pm$ 0.37	66.35	61.57 $\pm$ 0.30	216.46
Correlation	CorNet-ECM	97.71 $\pm$ 0.61	1.01	81.35 $\pm$ 1.27	0.60	92.17 $\pm$ 0.49	0.50	65.04 $\pm$ 0.14	12.06
	CorNet-LECM	98.40 $\pm$ 0.70	1.12	78.05 $\pm$ 1.14	0.64	91.17 $\pm$ 0.32	0.54	65.03 $\pm$ 0.10	12.68
	CorNet-OLM	97.57 $\pm$ 0.76	1.35	81.46 $\pm$ 0.61	0.93	91.63 $\pm$ 0.12	0.79	64.41 $\pm$ 0.23	16.07
	CorNet-LSM	96.24 $\pm$ 1.48	1.50	74.89 $\pm$ 1.07	0.98	83.43 $\pm$ 0.65	0.83	60.69 $\pm$ 0.85	16.28
	CorNet-PHCM	96.56 $\pm$ 0.86	2.37	82.26 $\pm$ 0.92	1.10	90.03 $\pm$ 0.63	0.77	60.01 $\pm$ 0.22	16.92

Table 5.20: Five-fold results and training time per epoch on four data sets. The top 3 results are highlighted with **red**, **blue**, and **cyan**. * denotes reproduced results due to missing official code.

GyroGr-Scaling [159], SPDNet [106], SPDNetBN [31], RResNet [118], LieBN [50], SPD MLR [51], Gyro [159], and GyroSPD++ [161].

5.4.5.1 Main Results

Tab. 5.20 reports the five-fold results comparing our CorNets against existing SPD and Grassmannian baselines. We summarize the key observations below.

- **Effectiveness.** CorNets consistently outperform both SPD and Grassmannian networks. Specifically, CorNets surpass the classic SPDNet by *5.15%*, *17.69%*, *6.58%*, and *13.79%* on four data sets, respectively, and outperform the best Grassmannian networks by *7.76%*, *19.07%*, *6.86%*, and *7.45%*. Despite not using BN or residual blocks, CorNets achieve superior performance compared to SPDNetBN, SPDNetLieBN, and RResNet. Notably, although CorNets share the same high-level architecture as GyroSPD++ (one manifold convolutional layer followed by one manifold MLR layer), CorNets exhibit better performance. These results highlight the effectiveness of correlation embedding and our method for constructing correlation networks.
- **Optimal Metric.** The optimal metric for CorNets varies across data sets, indicating that the choice of geometry is a critical hyperparameter in Riemannian networks. Our framework enables seamless switching among five correlation geometries in a consistent architecture, demonstrating the adaptability of our approach to different tasks.

5.4. Full-Rank Correlation Networks

Data Set		HDM05					FPHA				
Conv	MLR	ECM	LECM	OLM	LSM	PHCM	ECM	LECM	OLM	LSM	PHCM
	Conv	ECM	LECM	OLM	LSM	PHCM	ECM	LECM	OLM	LSM	PHCM
ECM		81.35 ± 1.27	73.38 ± 0.34	80.11 ± 0.77	78.54 ± 0.43	80.80 ± 0.54	92.17 ± 0.49	91.50 ± 0.21	91.67 ± 0.28	87.37 ± 1.14	91.97 ± 0.24
LECM		66.49 ± 1.13	78.05 ± 1.14	79.21 ± 1.23	73.61 ± 0.99	58.37 ± 2.24	87.90 ± 0.57	91.17 ± 0.32	90.25 ± 0.25	89.63 ± 0.31	86.09 ± 0.98
OLM		77.82 ± 0.48	76.56 ± 0.89	81.46 ± 0.61	80.77 ± 0.81	77.39 ± 1.29	92.17 ± 0.58	92.27 ± 0.78	91.63 ± 0.12	89.90 ± 0.67	91.83 ± 0.15
LSM		68.83 ± 1.19	70.41 ± 1.57	67.56 ± 1.52	74.89 ± 1.07	72.69 ± 3.56	78.97 ± 2.80	75.10 ± 1.15	82.25 ± 3.38	83.43 ± 0.65	78.97 ± 4.97
PHCM		81.16 ± 0.40	80.05 ± 0.45	81.96 ± 0.51	78.28 ± 0.64	82.26 ± 0.92	88.30 ± 0.81	79.80 ± 0.69	87.37 ± 0.72	86.63 ± 0.27	90.03 ± 0.63

Table 5.21: Ablations on mixed geometries. Each row shows the metric used for Convolution (Conv), and each column is the metric for MLR. The diagonal entries indicate configurations where both layers use the same metric. The best result in each row is bold.

- **Efficiency.** CorNets achieve efficiency comparable to or better than several baseline methods. The most efficient CorNet variant is based on ECM, owing to the simplest computations of ECM. Although GyroSPD++ uses the same architecture, CorNets achieve significantly greater efficiency, attributed to the heavy computational cost of the AIM-based computations in GyroSPD++ and the lightweight Riemannian computations on the correlation manifold. Particularly, on the largest NTU120 data set, CorNet-ECM and CorNet-LECM are the top two most efficient ones.

5.4.5.2 Ablations on Mixed Geometries

Our main experiments use the same metric for convolution and MLR. To evaluate mixed geometries, we assign different metrics to the two layers. Tab. 5.21 reports five-fold results on HDM05 and FPHA. Overall, consistent metrics yield the best accuracy.

5.4.5.3 Visualization

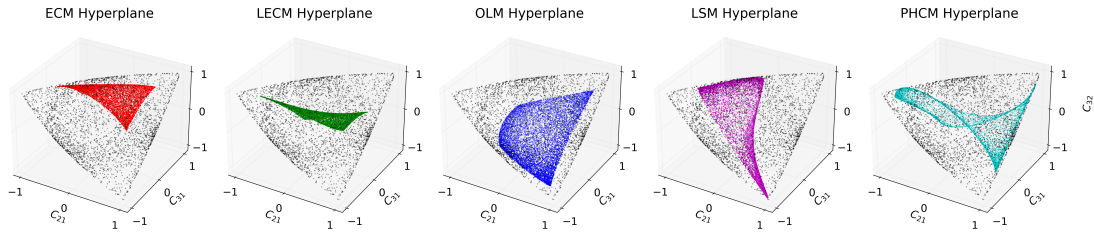


Figure 5.5: Illustration of the decision hyperplanes in the correlation MLRs under five different geometries. The 3×3 correlation manifold can be embedded as an open elliptope in $\mathbb{R}^3$, by visualizing the strictly lower triangular part of each $C \in \text{Cor}^+(3)$. The black dots denote the boundary. The PHCM hyperplane is defined by the one in the β -concatenated Poincaré space.

Fig. 5.5 shows that different metrics induce visibly distinct curved hyperplanes.

5.4.5.4 Potential and Necessity

Although correlation matrices are still SPD, naively treating them as SPD inputs and feeding them into existing SPD networks fails to leverage their intrinsic geometric structures. To illustrate

Input	Radar	HDM05	FPHA
SPD	93.25 $\pm$ 1.10	64.57 $\pm$ 0.61	85.59 $\pm$ 0.72
Correlation	89.49 $\pm$ 0.67	66.81 $\pm$ 0.73	83.37 $\pm$ 0.40

Table 5.22: SPDNet: SPD vs. correlation.

this, we use the classic SPDNet [106] but replace its covariance inputs with correlation matrices. The five-fold average results in Tab. 5.22 reveal two key insights: (1) on the HDM05 data set, correlation inputs lead to improved performance, suggesting that correlation embeddings can serve as compact and effective alternatives to covariance representations; and (2) on the other two data sets, the performance degrades, indicating that ignoring the specific geometry of correlation matrices can be detrimental. These findings highlight both the promise and the necessity of designing networks respecting the unique geometry of the correlation manifold.

5.4.5.5 Ablations on Correlation Embeddings

Data Set	Measurement	SPDMLR-Trivlz			CorMLR				
		LEM	LCM	AIM	ECM	LECM	OLM	LSM	PHCM
Radar	Acc	95.47 $\pm$ 0.66	95.55 $\pm$ 0.35	94.87 $\pm$ 0.87	89.47 $\pm$ 0.93	87.41 $\pm$ 0.23	85.79 $\pm$ 0.83	91.63 $\pm$ 0.32	83.33 $\pm$ 1.29
	Fit Time (s/epoch)	0.65	0.63	0.99	0.56	0.62	0.78	0.68	0.74
HDM05	Acc	54.31 $\pm$ 1.65	45.12 $\pm$ 1.05	52.46 $\pm$ 2.44	65.57 $\pm$ 0.62	64.44 $\pm$ 0.63	62.86 $\pm$ 0.65	64.01 $\pm$ 0.92	62.78 $\pm$ 0.85
	Fit Time (s/epoch)	3.24	5.38	260.67	3.18	3.87	3.39	3.57	2.73
FPHA	Acc	84.13 $\pm$ 1.14	76.62 $\pm$ 0.43	83.25 $\pm$ 0.59	85.37 $\pm$ 0.16	85.24 $\pm$ 0.22	84.67 $\pm$ 0.27	80.17 $\pm$ 0.15	73.67 $\pm$ 0.32
	Fit Time (s/epoch)	0.51	0.52	18.96	0.51	0.64	0.8	0.81	0.45

Table 5.23: Comparison of SPDMLR-Trivlz on raw covariances against CorMLR on raw correlations on all three data sets. The input matrix dimensions are 93×93 , 63×63 , and 20×20 , respectively.

To further evaluate the effectiveness of correlation embeddings, we compare the performance of directly classifying raw covariance matrices using the SPD MLRs in Thm. 117 with that of classifying corresponding raw correlation matrices using correlation MLR (CorMLR). The original SPDMLR involves an SPD matrix parameter for each class, which causes heavy Riemannian computations. For a fair comparison, we also implement a similar trivialization as Sec. 5.4.2.1, denoted as SPDMLR-Trivlz. We implement SPDMLR-Trivlz under LEM, LCM, and AIM, respectively. Tab. 5.23 presents the 5-fold average results on all three data sets. CorMLR performs better than SPDMLR-Trivlz on HDM05 and FPHA. Although CorMLR performs worse on Radar, we emphasize that

these comparisons are conducted on a single MLR layer, which fails to fully uncover the potential of correlation matrices. Besides, SPDMLR under AIM is much slower than others, especially on HDM05, due to its complex computations. In contrast, CorMLR, especially under ECM and PHCM, offers competitive or superior efficiency.

5.4.5.6 Analysis of Covariance versus Correlation

In this section, we analyze when and why correlation matrices provide stronger representations than covariance matrices. The coefficient of variation of diagonal variances quantifies the variability of diagonal variances via per-sample coefficients of variation, and the ratio of diagonal to off-diagonal entries compares the magnitudes of diagonal and off-diagonal entries via their ratios. These analyses lead to two insights: (1) large variability and magnitude of diagonal elements can act as nuisance noise for SPD networks by overshadowing informative off-diagonal correlations; (2) under such cases, correlation representations that normalize variances and emphasize pairwise correlations tend to be more effective, which is especially evident on HDM05.

Coefficient of Variation of Diagonal Variances.

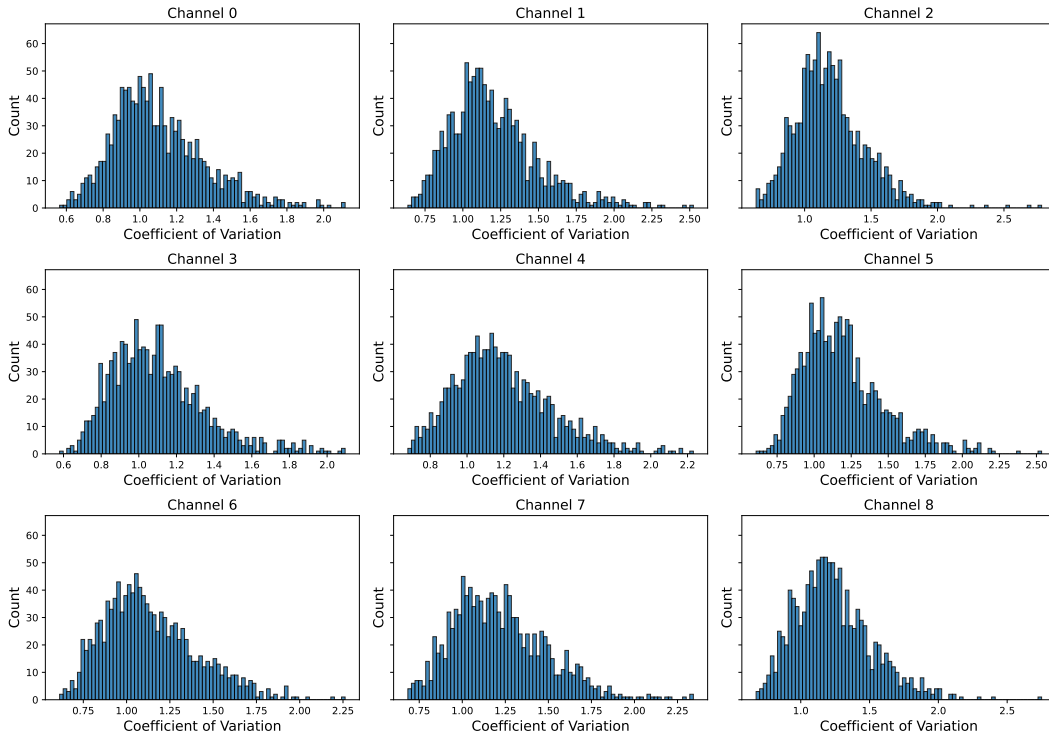


Figure 5.6: Distribution of per-sample coefficients of variation of diagonal variances on FPFA. Higher values indicate stronger diagonal variability, which could cause nuisance noise.

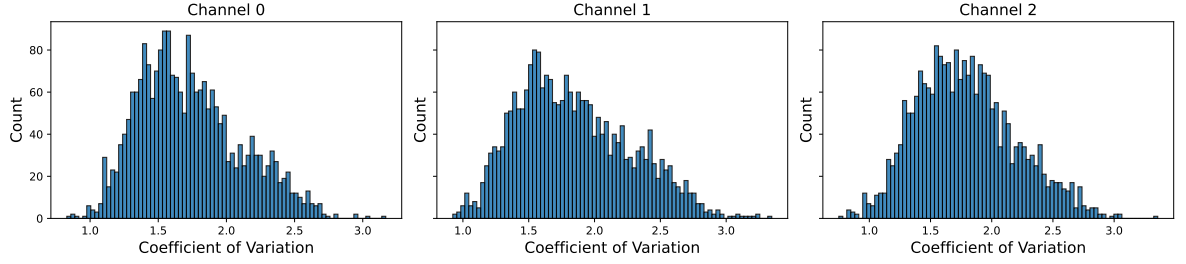


Figure 5.7: Distribution of per-sample coefficients of variation of diagonal variances on HDM05. Higher values indicate stronger diagonal variability, which could cause nuisance noise.

This section investigates why CorNets yield substantially larger gains over SPD networks on HDM05 compared to FPHA.

Setup. For each covariance matrix $\Sigma \in \mathcal{S}_{++}^n$ we extract the diagonal vector

$$v = (\Sigma_{11}, \dots, \Sigma_{nn}). \quad (5.82)$$

We compute the coefficient of variation of v as

$$\text{CV} = \frac{\text{std}(v)}{\text{mean}(v) + \varepsilon}, \quad (5.83)$$

where $\varepsilon = 10^{-8}$ ensures numerical stability. As shown in Sec. A.3.6.1, each sequence is modeled as a c -channel tensor of covariance matrices. The above procedure yields one coefficient of variation per channel for each sample. We visualize their empirical distributions per channel.

Analysis. Figs. 5.6 and 5.7 show that the coefficients of variation w.r.t. diagonal variance are large on both data sets. On FPHA, most values fall between 0.8 and 2.0. On HDM05, they are even larger, typically between 1.0 and 3.0. Such large fluctuations indicate that diagonal variances change substantially and could bring nuisance noise for SPD networks. In contrast, correlation matrices allow CorNets to focus on pairwise relationships. This explains the consistent improvements over SPD networks and the larger gains on HDM05.

Ratio of Diagonal to Off-Diagonal Entries in Covariance Features.

5.4. Full-Rank Correlation Networks

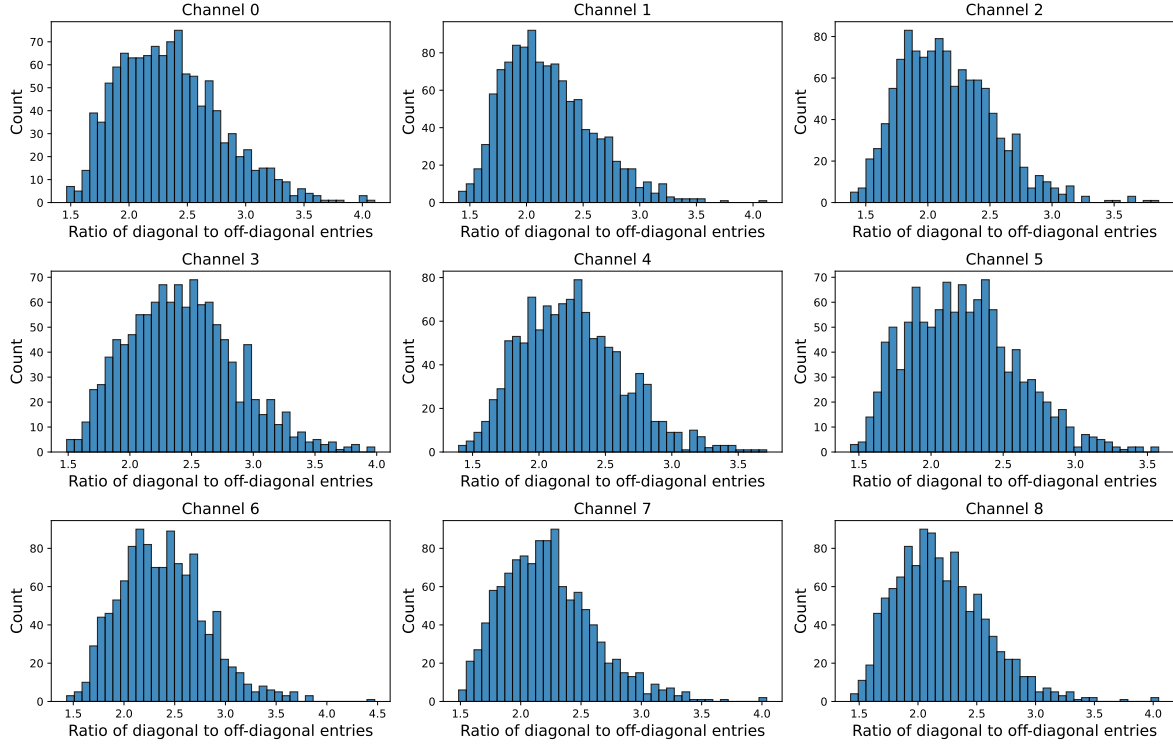


Figure 5.8: Distribution of ratios of diagonal to off-diagonal entries on FPHA.

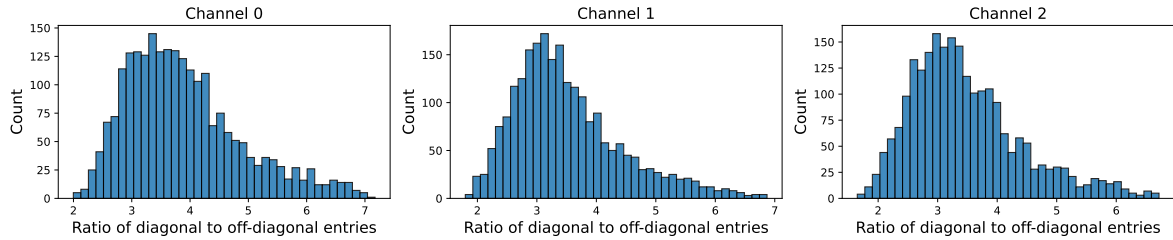


Figure 5.9: Distribution of ratios of diagonal to off-diagonal entries on HDM05.

This section further examines why CorNets achieve larger gains over SPD networks on HDM05 than on FPHA. We analyze the ratio of diagonal to off-diagonal entries in covariance matrices on FPHA and HDM05, to quantify how strongly variance terms overshadow pairwise correlations.

Setup. For each covariance matrix $\Sigma \in \mathcal{S}_{++}^n$ we compute the mean magnitude of diagonal entries

$$D = \frac{1}{n} \sum_{i=1}^n |\Sigma_{ii}|, \quad (5.84)$$

and the mean magnitude of off-diagonal entries

$$O = \frac{1}{n(n-1)} \sum_{i \neq j} |\Sigma_{ij}|. \quad (5.85)$$

We then form the sample-wise ratio

$$R = \frac{D}{O}, \quad (5.86)$$

which measures how much larger the diagonal amplitudes are compared to the off-diagonal correlations. Each sample yields one ratio per channel, and we visualize the empirical distributions of these ratios on FPHA and HDM05.

Analysis. Figs. 5.8 and 5.9 show that both data sets have ratios well above one. On FPHA, most ratios lie between 1.7 and 3.0, indicating that diagonal amplitudes are noticeably larger than off-diagonal correlations. HDM05 exhibits even larger ratios, typically between 2.0 and 6.0, with many above 3.0. These statistics indicate that covariance representations on both data sets are strongly dominated by diagonal entries, with more pronounced dominance on HDM05. When diagonal terms dominate, SPD networks trained on covariance inputs tend to overemphasize variances and underexploit informative pairwise correlations. Correlation matrices normalize variances and highlight off-diagonal interactions, which explains why CorNets outperform SPD baselines on both data sets and why the improvement is substantially larger on HDM05.

5.4.5.7 Normalized Covariance vs. Correlation

Setup. We evaluate SPD-based baselines by covariance inputs normalized by their largest eigenvalue. Given a covariance matrix Σ , we get the normalized SPD input $\hat{\Sigma} = \Sigma / \lambda_{\max}(\Sigma)$ and feed it into existing SPD networks. This variant is denoted by “-EigN”. We report results on the Radar, HDM05, and FPHA data sets for representative SPD models: SPDNet, SPDNetBN, SPDResNet, SPDNetLieBN, SPDNetMLR, GyroAI, and GyroSPD++. Here, SPDResNet is implemented under the LEM, while SPDNetLieBN follows the LCM.

5.4. Full-Rank Correlation Networks

Manifold	Method	Radar	HDM05	FPHA
$\mathcal{S}_{++}^n$	SPDNet	93.25 ± 1.10	64.57 ± 0.61	85.59 ± 0.72
	SPDNet-EigN	86.91 ± 0.57	66.62 ± 0.73	84.90 ± 0.62
	SPDNetBN	94.85 ± 0.99	71.28 ± 0.79	89.33 ± 0.49
	SPDNetBN-EigN	89.25 ± 1.19	71.59 ± 0.68	88.47 ± 0.39
	SPDResNet	95.89 ± 0.86	70.12 ± 2.45	85.07 ± 0.99
	SPDResNet-EigN	92.61 ± 0.96	71.02 ± 0.91	84.53 ± 0.46
	SPDNetLieBN	94.80 ± 0.71	71.78 ± 0.44	86.33 ± 0.43
	SPDNetLieBN-EigN	88.91 ± 1.21	70.61 ± 1.04	83.73 ± 0.65
	SPDNetMLR	94.59 ± 0.82	65.90 ± 0.93	85.60 ± 0.43
	SPDNetMLR-EigN	89.41 ± 0.58	66.89 ± 0.63	83.63 ± 1.09
$\text{Cor}^+(n)$	GyroAI	96.29 ± 0.48	72.34 ± 1.06	89.60 ± 0.37
	GyroAI-EigN	91.36 ± 0.80	72.64 ± 0.70	89.90 ± 0.31
	GyroSPD++	95.20 ± 0.88	69.82 ± 1.79	89.50 ± 0.37
	GyroSPD++-EigN	90.83 ± 1.09	66.92 ± 0.28	84.29 ± 0.14
	CorNet-ECM	97.71 ± 0.61	81.35 ± 1.27	92.17 ± 0.49
	CorNet-LECM	98.40 ± 0.70	78.05 ± 1.14	91.17 ± 0.32
	CorNet-OLM	97.57 ± 0.76	81.46 ± 0.61	91.63 ± 0.12
	CorNet-LSM	96.24 ± 1.48	74.89 ± 1.07	83.43 ± 0.65
	CorNet-PHCM	96.56 ± 0.86	82.26 ± 0.92	90.03 ± 0.63

Table 5.24: SPD networks with or without normalized SPD inputs.

Results. Tab. 5.24 summarizes the results. On HDM05, eigenvalue normalization has only a marginal effect and the normalized variants achieve accuracy comparable to their unnormalized counterparts. On FPHA and, in particular, on Radar, normalization usually reduces accuracy. The behavior of GyroSPD++ is especially informative. GyroSPD++ and CorNet share a similar architecture, consisting of one convolution followed by an MLR layer. However, GyroSPD++-EigN performs worse than GyroSPD++ on all three data sets, while CorNet with correlation inputs achieves clear improvements over GyroSPD++. These phenomena can be explained by two factors.

- (1) **Redundancy.** The raw samples on HDM05 and FPHA have already undergone centering, scaling, and normalization before covariance modeling. Dividing by $\lambda_{\max}(\Sigma)$ therefore introduces little additional control over scale, which explains the marginal effect on HDM05.
- (2) **Scaled Covariance versus Correlation.** Since EigN is equivalent to uniformly rescaling the raw samples before covariance computation, the normalized covariance matrices remain covariances and do not encode new statistical information.

Moreover, forcing the largest eigenvalue to 1 can remove potentially informative differences in overall energy across samples, which aligns with the degradation observed for EigN variants, especially GyroSPD++-EigN. In contrast, correlation normalization uses a different scaling factor for each pair of variables,

$$\text{Cor}_{ij} = \frac{\Sigma_{ij}}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}, \quad (5.87)$$

producing standardized correlation coefficients. Therefore, global eigenvalue scaling is statistically distinct from correlation normalization and fails to capture the benefits of explicit correlation modeling.

5.4.5.8 Ablations on Activations

Metric	Activation	Radar	HDM05	FPHA
ECM	ReLU	97.41 $\pm$ 0.25	81.23 $\pm$ 0.46	89.80 $\pm$ 0.58
	None	97.71 $\pm$ 0.61	81.35 $\pm$ 1.27	92.17 $\pm$ 0.49
LECM	ReLU	97.23 $\pm$ 0.67	77.51 $\pm$ 1.02	91.00 $\pm$ 0.15
	None	98.40 $\pm$ 0.70	78.05 $\pm$ 1.14	91.17 $\pm$ 0.32
OLM	ReLU	97.52 $\pm$ 0.47	81.86 $\pm$ 0.65	91.47 $\pm$ 0.19
	None	97.57 $\pm$ 0.76	81.46 $\pm$ 0.61	91.63 $\pm$ 0.12
LSM	ReLU	95.60 $\pm$ 0.97	N/A	N/A
	None	96.24 $\pm$ 1.48	74.89 $\pm$ 1.07	83.43 $\pm$ 0.65
PHCM	ReLU	96.40 $\pm$ 0.25	77.32 $\pm$ 1.56	88.63 $\pm$ 0.22
	None	96.56 $\pm$ 0.86	82.26 $\pm$ 0.92	90.03 $\pm$ 0.63

Table 5.25: Comparison of CorNet with or without activations.

In the main experiments, we follow HNN++ [181] and GyroSPD++ [161], and do not use explicit activations, as the manifold itself introduces nonlinearity. We further conduct an ablation on activations. Following Ganea et al. [76, Sec. 3.2], we define activations in the tangent space at the identity, *i.e.*, $\text{Exp}_I \circ \delta \circ \text{Log}_I$ for four Log-Euclidean metrics, and $\text{Exp}_0 \circ \delta \circ \text{Log}_0$ for PHCM in the β -concatenated Poincaré vector, where δ is ReLU [82]. Specifically, we insert a ReLU after the correlation convolution. As shown in Tab. 5.25, adding activations generally yields no benefits and can even degrade performance. The variant without activation consistently achieves higher or comparable accuracy, except CorNet-OLM for HDM05. Moreover, CorNet-LSM with activation

fails to converge on HDM05 and FPHA. These results suggest that CorNet already provides sufficient nonlinearity, rendering additional activations redundant.

5.4.5.9 Scalability of Correlation Metrics

Dim	ECM	LECM	OLM	LSM	PHCM
30	0.0004	0.0018	0.0012	0.0019	0.0131
50	0.0004	0.0027	0.0318	0.0334	0.0211
100	0.0008	0.0054	0.0764	0.0781	0.0413
150	0.0015	0.0100	0.1247	0.1267	0.2284
200	0.0025	0.0197	0.1906	0.1938	0.3320
250	0.0037	0.0345	0.2352	0.2379	0.4414
300	0.0053	0.0733	0.3434	0.3454	0.5732
400	0.0092	0.1796	0.5163	0.5261	0.4807
500	0.0143	0.3076	0.6907	0.6961	0.5693
600	0.0206	0.5983	0.9331	0.9484	0.7923
700	0.0289	1.0961	1.2432	1.2575	1.0417
800	0.039	1.8689	1.6658	1.6815	1.3387
900	0.0535	2.9886	2.2156	2.2303	1.7324
1000	0.0706	3.7259	2.539	2.5783	1.229

Table 5.26: Average runtime (s) of a single forward pass in CorNet under different metrics and input dimensions. The best results are **bold**.

We evaluate the computational efficiency of correlation metrics across increasing input dimensions using CorNet with one correlation FC layer followed by one correlation MLR layer. Each input correlation matrix of size $[n, n]$ is mapped to $[20, 20]$ by the FC layer and then classified into 10 classes by the MLR layer. For each listed dimension $30 \leq n \leq 1000$, we randomly generate 30 correlation matrices and record the average runtime of a single forward pass. As implied by Tab. 2.7, the runtime is governed by two factors: the codomain computation (Euclidean or hyperbolic) and the complexity of the diffeomorphism. The results are summarized in Tab. 5.26. We have the following findings.

- ECM is consistently the most efficient metric, benefiting from both a Euclidean codomain and the simplest diffeomorphism.

- At very low dimensions ($n \leq 100$), the relative costs of the non-ECM metrics are not yet stable. At $n = 50$ and $n = 100$, the ordering is

$$\text{ECM} < \text{LECM} < \text{PHCM} < \text{OLM} \approx \text{LSM}. \quad (5.88)$$

At the smallest tested dimension $n = 30$, all runtimes remain small and their relative ordering differs. At intermediate dimensions ($150 \leq n \leq 300$), the ordering becomes

$$\text{ECM} < \text{LECM} < \text{OLM} \approx \text{LSM} < \text{PHCM}. \quad (5.89)$$

Here, the cost of PHCM’s hyperbolic computations dominates, while the dimension-dependent cost of LECM’s matrix functions is not yet pronounced.

- From $n = 400$, PHCM becomes faster than OLM and LSM, and at $n = 700$, it also becomes faster than LECM. At high dimensions ($n \geq 800$), the ordering is

$$\text{ECM} < \text{PHCM} < \text{OLM} \approx \text{LSM} < \text{LECM}. \quad (5.90)$$

Here, diffeomorphisms dominate: ECM and PHCM scale better thanks to relatively lightweight Cholesky decomposition, while OLM and LSM slow down due to matrix logarithm/exponentiation. LECM is the slowest, as its $\log \circ \Theta$ requires two nested matrix functions.

5.5 Conclusion

This chapter developed manifold-specific neural components and architectures by exploiting the additional structures of particular hyperbolic models and correlation manifolds.

The first part addressed representation choice through PVNN. The unconstrained PV model is connected to the Poincaré and hyperboloid models by Riemannian isometries, but it avoids their explicit constraints. By deriving closed-form core Riemannian operators and relating them to the PV gyrovector structure, we constructed PV MLR, FC, convolutional, activation, and normalization layers. Together, these layers constitute a complete PVNN framework for constructing model-specific hyperbolic architectures. Experiments demonstrate the superior numerical stability and competitive performance of the PV representation.

The second part shifted attention from the representation itself to the geometric principle for building neural layers. Using Busemann functions and horospheres, we developed a common construction for the Poincaré and Lorentz models. BMLR ad-

mits an exact point-to-horosphere interpretation, avoids an additional manifold-valued point parameter, supports batch-efficient evaluation, and recovers Euclidean MLR in the zero-curvature limit. The same Busemann logits yield explicit BFC layers with practical $\mathcal{O}(nm)$ complexity and corresponding Euclidean limits. Experiments on image classification, genome sequence learning, node classification, and link prediction showed that these layers generally improve upon existing hyperbolic alternatives while retaining comparable computational cost.

The third part extended manifold-specific network design from hyperbolic vectors to full-rank correlation matrices. The four flat geometries, ECM, LECM, OLM, and LSM, admit Euclidean isometries that yield closed-form correlation MLR, FC, and convolutional layers. For the non-flat PHCM geometry, the Cholesky representation identified correlation matrices with a product of Poincaré balls, enabling analogous layers through β -concatenation and β -splitting. Analytic gradients for the Riemannian computations enabled accurate end-to-end backpropagation under OLM and LSM. Across the four evaluated data sets, the best CorNet geometry outperformed the considered SPD and Grassmannian baselines.

All three designs nevertheless operate under prescribed Riemannian geometries. The next chapter therefore turns from the design of modules and architectures to that of the underlying metrics themselves.

Chapter 6

Fast and Stable Geometries on SPD Manifolds

6.1 Introduction

The preceding chapters developed Riemannian neural components and architectures under prescribed Riemannian metrics. Despite their different constructions, they share the underlying metric as a common starting point.

As reviewed in Chapter 2, a Riemannian metric assigns inner products to tangent spaces and determines distance, geodesics, exponential and logarithmic maps, and parallel transport. The induced distance also defines the Fréchet statistics used to summarize manifold-valued features. When combined with compatible Lie group or gyrovector structures, a Riemannian metric further supports manifold analogues of addition and scalar multiplication. These geometric primitives offer powerful toolkits for building Riemannian neural networks.

Based on this observation, we shift our attention from designing neural modules and architectures under prescribed geometries to designing the underlying geometry itself. Our goal is to balance geometric flexibility, theoretical convenience, computational efficiency, and numerical stability. Accordingly, we seek geometries whose Riemannian operators and compatible algebraic operations admit closed-form expressions and can be inserted directly into deep learning architectures. We focus on the SPD manifold, which encodes covariance or other second-order statistics and arises naturally in many applications [37, 31, 134, 61, 152, 141, 231, 106].¹

¹Our focus is distinct from SPD metric-learning methods that learn distance functions induced by an existing Riemannian metric. Instead, we seek to design new Riemannian metrics themselves.

We pursue two complementary routes. Sec. 6.2 starts from pullback Euclidean geometry and proposes Adaptive Log-Euclidean Metrics (ALEMs). The resulting adaptive geometry retains closed-form Riemannian operators and a convenient abelian Lie group structure. Sec. 6.3 instead exposes the product structure of the Cholesky manifold and transfers these geometries to the SPD manifold, yielding simple and stable closed-form Riemannian and gyro operators.

6.2 Adaptive Log-Euclidean Metrics

6.2.1 Introduction

As reviewed in Sec. 2.9.1, most popular Riemannian metrics on the SPD manifold are fixed, which can limit the expressive capacity of the associated geometry. A common way to construct SPD metrics is through pullbacks along diffeomorphisms, which transfer Riemannian structures from simpler source manifolds. For instance, Thanwerdas and Pennec [195] explained AIM as the pullback metric from a left-invariant metric on the Cholesky manifold. The matrix-power deformations in Secs. 3.2.5.1 and 4.3.3.1 are also representative examples.

Inspired by the above observations, we leverage pullback techniques to introduce adaptive Riemannian metrics. In particular, we first show that several Riemannian metrics on SPD manifolds, including LEM, LCM, and their generalizations, can be explained as pullback metrics from the standard Euclidean space. We refer to these metrics as pullback Euclidean metrics. Then, we propose a general framework for characterizing the properties of pullback Euclidean metrics. Our framework can explain the widely used LEM [9] and LCM [137]. We focus on LEM on SPD manifolds and extend it into *Adaptive Log-Euclidean Metrics (ALEMs)*. Besides, we present a complete theoretical discussion on the properties of ALEMs. Compared with the existing Riemannian metrics, our metrics are learnable, adapting to the characteristics of the data sets. The effectiveness of our metrics is demonstrated by experiments as well as the applications to recently developed Riemannian building blocks, including the RBN framework developed in Chapter 3, Riemannian residual blocks [117], and the Riemannian classifiers developed in Chapter 4. Drawing on this, our **contributions** are summarized as follows:

- (1) We reveal the connection of two popular Riemannian metrics (LEM and LCM) by the pullback technique and propose a general framework for pullback Euclidean metrics.

- (2) Based on our framework, we propose specific ALEMs on SPD manifolds and conduct comprehensive analyses in terms of the algebraic, analytic, and geometric properties.
- (3) Extensive experiments on widely used SPD learning benchmarks demonstrate that our metrics exhibit consistent performance gain across data sets.²

Outline. The necessary background on differential geometry, pullback metrics, and SPD geometry has been established in Chapter 2. Sec. 6.2.2 develops ALEM and its differentials, and Sec. 6.2.2.5 studies its geometric properties. Sec. 6.2.3 derives the gradients and parameter-update rules for the general matrix logarithm and exponential. Sec. 6.2.4 instantiates the general matrix logarithm as ALog in SPDNet and applies ALEM to other Riemannian building blocks. Proofs are deferred to Sec. B.9.

6.2.2 Adaptive Log-Euclidean Metrics

In this section, we show that both (α, β) -LEM and LCM are pullback metrics from the Euclidean space. Inspired by this observation, we present a general framework for characterizing pullback Euclidean metrics. Then, we focus on generalizing LEM.

6.2.2.1 Rethinking (α, β) -LEM and LCM

Among the existing Riemannian metrics on the SPD manifold, LEM is popular in many applications, given its closed form for the Fréchet mean and clear vector-space and Lie-group structures. In addition, the nascent LCM, which is gaining increasing attention, shares similar properties with LEM. LEM is derived from Lie group translation [9], while LCM is obtained as the pullback of a metric on $\mathcal{L}_{++}^n$ [137]. Besides, (α, β) -LEM is obtained as a pullback of LEM [196]. However, the same mathematical logic underlies their derivations. We denote the Euclidean space of $n \times n$ lower triangular matrices by LT^n . We define $\psi_{\text{LC}} : \mathcal{S}_{++}^n \rightarrow \text{LT}^n$ as

$$\psi_{\text{LC}}(P) = \lfloor L \rfloor + \text{Dlog}(\mathbb{D}(L)), \quad (6.1)$$

where L is the Cholesky factor of the SPD matrix P , $\lfloor L \rfloor$ is the strictly lower part of L , $\mathbb{D}(L)$ is a diagonal matrix with diagonal elements of L , and Dlog applies the natural logarithm element-wise to the diagonal. Then, we have the following theorem.

²The code is available at <https://github.com/GitZH-Chen/ALEM>.

Theorem 147. $\downarrow$ (α, β) -LEM is the pullback metric from the Euclidean space of $\mathcal{S}^n$ with an $O(n)$ -invariant inner product $\langle \cdot, \cdot \rangle^{(a,b)}$ by matrix logarithm. Specifically, the standard LEM is the pullback metric from the Euclidean space of $\mathcal{S}^n$ with the standard Frobenius inner product by matrix logarithm. LCM is the pullback metric from LT^n with the Frobenius inner product by ψ_{LC} .

As Euclidean spaces of the same dimension are naturally isometric, it follows that both (α, β) -LEM and LCM are pulled back from the standard Euclidean space $\mathcal{S}^n$.

Corollary 148. $\downarrow$ (α, β) -LEM and LCM are pullback metrics from $\mathcal{S}^n$ with the standard Frobenius inner product.

6.2.2.2 Pullback Euclidean Metrics on SPD Manifolds

Sec. 6.2.2.1 has shown how LEM is derived from matrix logarithm. Besides, as shown in Arsigny et al. [9], operations in Lie group and linear space on $\mathcal{S}_{++}^n$ are also induced from matrix logarithm. Now, let us explain the underlying mechanism in detail. A matrix logarithm is a diffeomorphism (a smooth bijection with a smooth inverse). The property of bijection offers the possibility of transferring algebraic structures from $\mathcal{S}^n$ into $\mathcal{S}_{++}^n$. The smoothness of the matrix logarithm and its inverse suggests that smooth structures, such as a Lie group structure and a Riemannian metric, can be transferred to $\mathcal{S}_{++}^n$. More generally, given an arbitrary diffeomorphism $\phi : \mathcal{S}_{++}^n \rightarrow \mathcal{S}^n$, it suffices to pull various properties from the Euclidean space back to the SPD manifold $\mathcal{S}_{++}^n$ by ϕ as well. Besides, the computation of the induced operators in $\mathcal{S}_{++}^n$ by ϕ is usually simple.

Lemma 149. $\downarrow$ Let $S_1, S_2 \in \mathcal{S}_{++}^n$, $V \in T_{S_1}\mathcal{S}_{++}^n$, and $k \in \mathbb{R}$, and let g^E be the Frobenius inner product in $\mathcal{S}^n$. Let $\phi : \mathcal{S}_{++}^n \rightarrow \mathcal{S}^n$ be a diffeomorphism, and denote its differential at $S \in \mathcal{S}_{++}^n$ by $\phi_{*,S}$. We define the following operations:

$$\text{Element Addition: } S_1 \odot_\phi S_2 = \phi^{-1}(\phi(S_1) + \phi(S_2)), \quad (6.2)$$

$$\text{Scalar Multiplication: } k \otimes_\phi S_2 = \phi^{-1}(k\phi(S_2)), \quad (6.3)$$

$$\text{Inner Product: } \langle S_1, S_2 \rangle_\phi = \langle \phi(S_1), \phi(S_2) \rangle, \quad (6.4)$$

$$\text{Riemannian Metric: } g^\phi = \phi^* g^E, \quad (6.5)$$

Then, we have the following conclusions:

(1) $\{\mathcal{S}_{++}^n, \odot_\phi, \otimes_\phi, \langle \cdot, \cdot \rangle_\phi\}$ is a Hilbert space over $\mathbb{R}$.

(2) $\{\mathcal{S}_{++}^n, \odot_\phi\}$ is an abelian Lie group. $\{\mathcal{S}_{++}^n, g^\phi\}$ is a Riemannian manifold. The associated Riemannian operators are as follows:

$$d^\phi(S_1, S_2) = \|\phi(S_1) - \phi(S_2)\|_F, \quad (6.6)$$

$$\text{Exp}_{S_1} V = \phi^{-1}(\phi(S_1) + \phi_{*,S_1} V), \quad (6.7)$$

$$\text{Log}_{S_1} S_2 = \phi_{*,\phi(S_1)}^{-1}(\phi(S_2) - \phi(S_1)), \quad (6.8)$$

$$\text{PT}_{S_1 \rightarrow S_2}(V) = \phi_{*,\phi(S_2)}^{-1} \circ \phi_{*,S_1}(V), \quad (6.9)$$

where $\|\cdot\|_F$ is the Frobenius norm, $V \in T_{S_1}\mathcal{S}_{++}^n$ is a tangent vector, Exp_{S_1} , Log_{S_1} , and $\text{PT}_{S_1 \rightarrow S_2}$ are the Riemannian exponential map at S_1 , logarithmic map at S_1 , and parallel transport along the geodesic connecting S_1 and S_2 , respectively, and ϕ_*^{-1} denotes the differential of ϕ^{-1} . Then g^ϕ is a bi-invariant metric, called a Pullback Euclidean Metric induced by ϕ .

(3) ϕ is an isomorphism: (a) a linear isomorphism preserving the inner product; (b) a Lie group isomorphism; (c) a Riemannian isometry.

In fact, (α, β) -LEM and LCM are special cases of Thm. 149, as are the linear-space and Lie-group structures in Arsigny et al. [9] and the Lie-group structure in Lin [137]. In addition, neither Arsigny et al. [9] nor Lin [137] reveals the Hilbert space structures in $\mathcal{S}_{++}^n$.

6.2.2.3 Adaptive Log-Euclidean Metrics

The key to Thm. 149 lies in the diffeomorphism ϕ . If we have a proper ϕ , Riemannian metrics on SPD manifolds can be induced. In the following, we will present our mappings and then discuss the induced metrics.

As reviewed in Sec. 2.8.1, the matrix logarithm reduces to a scalar logarithm, which is a diffeomorphism between $\mathbb{R}_{++}$ and $\mathbb{R}$. Following this hint, the eigenvalue-based diffeomorphism between $\mathcal{S}_{++}^n$ and $\mathcal{S}^n$ reduces to a scalar diffeomorphism between $\mathbb{R}_{++}$ and $\mathbb{R}$. A very natural idea is to substitute the natural logarithm with logarithms with arbitrary proper bases. Throughout this part, $\log(\cdot)$ without a subscript denotes the natural scalar or matrix logarithm. Every scalar or matrix logarithm with a general or adaptive base is written explicitly as $\log_\alpha(\cdot)$, without omitting the subscript. The base parameter α is interpreted as either a scalar or a vector according to its argument. For

a scalar base $\alpha \in \mathbb{R}_{++} \setminus \{1\}$ and $x \in \mathbb{R}_{++}$, we define

$$\log_{\alpha}(x) = \frac{\log(x)}{\log(\alpha)}. \quad (6.10)$$

When $\alpha = e$, this scalar logarithm reduces to the natural logarithm, which we write without a subscript as $\log(\cdot)$. For a diagonal matrix X , the same notation is extended to a base vector as

$$\log_{\alpha}(X) = \text{diag}(\log_{a_1}(x_{11}), \log_{a_2}(x_{22}), \dots, \log_{a_n}(x_{nn})), \quad (6.11)$$

where $\alpha = (a_1, a_2, \dots, a_n) \in (\mathbb{R}_{++} \setminus \{1\})^n$ is the base vector, $\text{diag}(\cdot)$ is the diagonalization operator, and X is an $n \times n$ diagonal matrix. When α is scalar in a matrix expression, it denotes the constant base vector $(\alpha, \dots, \alpha)$. Together with eigendecomposition, a general matrix logarithm is defined by

$$\log_{\alpha}(S) = U \log_{\alpha}(\Sigma) U^{\top}, \quad (6.12)$$

where $S = U \Sigma U^{\top}$ is the eigendecomposition. As a special case,

$$\alpha = (e, e, \dots, e) \implies \log_{\alpha} = \log. \quad (6.13)$$

As with the scalar logarithm, we have the following proposition.

Proposition 150 (Diffeomorphism). $\Downarrow$ $\log_{\alpha}$ is a diffeomorphism, a smooth bijection with a smooth inverse $\log_{\alpha}^{-1} : \mathcal{S}^n \rightarrow \mathcal{S}_{++}^n$ defined as

$$\log_{\alpha}^{-1}(X) = U \text{diag}(a_1^{\Sigma_{11}}, a_2^{\Sigma_{22}}, \dots, a_n^{\Sigma_{nn}}) U^{\top}, \quad (6.14)$$

where $X = U \Sigma U^{\top}$ is the eigendecomposition.

Remark 151. The general matrix logarithm $\log_{\alpha}$ is an arbitrary member of the following family

$$\{\log_{\alpha} \mid \alpha = (a_1, \dots, a_n) \in (\mathbb{R}_{++} \setminus \{1\})^n\}. \quad (6.15)$$

Besides, there could be some ambiguity in Eq. (6.12) under different arrangements of eigenvalues and eigenvectors. In fact, there is a correspondence between scalar

$\log_{a_i}$ and eigenvalues and eigenvectors. See Sec. A.4.4.1 for more details.

Since $\log_\alpha$ is a diffeomorphism from $\mathcal{S}_{++}^n$ onto $\mathcal{S}^n$, all the results in Thm. 149 hold true.

Theorem 152. [↓] *Following the notation in Thm. 149, we define $\oplus^{\text{ALE}}$ and $\odot^{\text{ALE}}$ as in Eqs. (6.2) and (6.3). We define $\langle \cdot, \cdot \rangle_{\log_\alpha}$ and $g^{\log_\alpha}$ as in Eqs. (6.4) and (6.5). Then, we have the following conclusions:*

- (1) $\{\mathcal{S}_{++}^n, \oplus^{\text{ALE}}, \odot^{\text{ALE}}, \langle \cdot, \cdot \rangle_{\log_\alpha}\}$ is a Hilbert space over $\mathbb{R}$.
- (2) $\{\mathcal{S}_{++}^n, \oplus^{\text{ALE}}\}$ is an abelian Lie group. $g^{\log_\alpha}$ is a Riemannian metric on $\mathcal{S}_{++}^n$. We call this metric the Adaptive Log-Euclidean Metric (ALEM) and denote $g^{\log_\alpha}$ by g^{ALE} . The associated Riemannian operators are as follows:

$$d^{\text{ALE}}(S_1, S_2) = \|\log_\alpha(S_1) - \log_\alpha(S_2)\|_{\text{F}}, \quad (6.16)$$

$$\text{Exp}_{S_1} V = \log_\alpha^{-1} \left(\log_\alpha(S_1) + (\log_\alpha)_{*, S_1} V \right), \quad (6.17)$$

$$\text{Log}_{S_1} S_2 = (\log_\alpha^{-1})_{*, X_1} (\log_\alpha(S_2) - \log_\alpha(S_1)), \quad (6.18)$$

$$\text{PT}_{S_1 \rightarrow S_2}(V) = (\log_\alpha^{-1})_{*, X_2} \circ (\log_\alpha)_{*, S_1}(V), \quad (6.19)$$

where $X_i = \log_\alpha(S_i) \in \mathcal{S}^n$ for $i = 1, 2$.

- (3) $\log_\alpha$ is an isomorphism: (a) a linear isomorphism preserving the inner product; (b) a Lie group isomorphism; (c) a Riemannian isometry.

Remark 153. Obviously, ALEM varies with different base parameters α in $\log_\alpha$. We thus use the plural to describe our metrics. Besides, our metrics can be learned. This is why we call them adaptive metrics.

Analogously to (α, β) -LEM, we can also define (a, b) -ALEM as the pullback of an $\text{O}(n)$ -invariant inner product:

$$g^{(a,b)\text{-ALE}} = \log_\alpha^* g^{(a,b)\text{-E}}, \quad (6.20)$$

where we denote the $\text{O}(n)$ -invariant inner product $\langle \cdot, \cdot \rangle^{(a,b)}$ by $g^{(a,b)\text{-E}}$. $g^{(a,b)\text{-ALE}}$ also shares the properties presented in Thm. 152. Nevertheless, this part focuses on $(a, b) = (1, 0)$.

6.2.2.4 Differentials of General Logarithms

Eqs. (6.17) to (6.19) require the differential maps of $\log_\alpha$ and $\log_\alpha^{-1}$. This subsection introduces the concrete formulae of the associated differential maps.

Proposition 154 (Differentials). $\Downarrow$ For a tangent vector $V \in T_S \mathcal{S}_{++}^n$, the differential $(\log_\alpha)_{*,S} : T_S \mathcal{S}_{++}^n \rightarrow T_{\log_\alpha(S)} \mathcal{S}^n$ of $\log_\alpha$ at $S \in \mathcal{S}_{++}^n$ is given by

$$(\log_\alpha)_{*,S}(V) = Q + Q^\top + W, \quad (6.21)$$

where $Q = D_U \log_\alpha(\Sigma) U^\top$,

$$D_U = ((\sigma_1 I_n - S)^+ V u_1 \quad \cdots \quad (\sigma_n I_n - S)^+ V u_n),$$

$$W = U \operatorname{diag} \left(\frac{u_1^\top V u_1}{\sigma_1 \log(a_1)}, \dots, \frac{u_n^\top V u_n}{\sigma_n \log(a_n)} \right) U^\top,$$

$(\cdot)^+$ is the Moore–Penrose inverse, $u_1, \dots, u_n$ are orthonormal eigenvectors of S , and the associated eigenvalues are $\sigma_1, \dots, \sigma_n$.

Symmetrically, for a tangent vector $\tilde{V} \in T_X \mathcal{S}^n$, the differential $(\log_\alpha^{-1})_{*,X} : T_X \mathcal{S}^n \rightarrow T_{\log_\alpha^{-1}(X)} \mathcal{S}_{++}^n$ of $\log_\alpha^{-1}$ at $X \in \mathcal{S}^n$ is given by

$$(\log_\alpha^{-1})_{*,X}(\tilde{V}) = \tilde{Q} + \tilde{Q}^\top + \tilde{W}, \quad (6.22)$$

where $X = \tilde{U} \tilde{\Sigma} \tilde{U}^\top$ is the eigendecomposition. Here, $D_{\tilde{U}}$ is defined similarly and $\tilde{Q} = D_{\tilde{U}} \operatorname{diag} (a_1^{\tilde{\sigma}_1}, \dots, a_n^{\tilde{\sigma}_n}) \tilde{U}^\top$. Moreover,

$$\tilde{W} = \tilde{U} \operatorname{diag} \left(\log(a_1) a_1^{\tilde{\sigma}_1} \tilde{u}_1^\top \tilde{V} \tilde{u}_1, \dots, \log(a_n) a_n^{\tilde{\sigma}_n} \tilde{u}_n^\top \tilde{V} \tilde{u}_n \right) \tilde{U}^\top.$$

Arsigny et al. [9] write the differential of the matrix exponential as an infinite series. The differential of $\log_\alpha^{-1}$ can also be rewritten in this way.

Proposition 155 (Differential as Infinite Series). $\Downarrow$ Following the notation in Thm. 154, the differential of $\log_\alpha^{-1}$ can also be formulated as

$$(\log_\alpha^{-1})_{*,X}(\tilde{V}) = \sum_{k=1}^{\infty} \frac{1}{k!} \left(\sum_{l=0}^{k-1} (\tilde{P}X)^{k-l-1} (D_{\tilde{P}}X + \tilde{P}\tilde{V})(\tilde{P}X)^l \right), \quad (6.23)$$

where $\tilde{P} = \tilde{U}B\tilde{U}^\top$, $B = \text{diag}(\log(a_1), \dots, \log(a_n))$, $D_{\tilde{P}} = D_{\tilde{U}}B\tilde{U}^\top + \tilde{U}BD_{\tilde{U}}^\top$.

When $\log_\alpha^{-1}$ is reduced to the matrix exponential, Eq. (6.23) coincides with the expression in Arsigny et al. [9, Eq. (8)], and our ALEM becomes exactly LEM.

6.2.2.5 Properties of ALEM

Since our ALEMs are natural generalizations of LEM, they intuitively share many of its properties. This subsection introduces some useful properties of our ALEMs for machine learning.

Fréchet means are important tools for SPD matrix learning [94, 36, 31, 34]. Like LEM, our ALEM also admits closed-form expressions for Fréchet means. We present a more general result, the weighted Fréchet mean.

Proposition 156 (Weighted Fréchet Means). [↓] For m points $S_1, \dots, S_m$ on the SPD manifold with associated weights $w_1, \dots, w_m \in \mathbb{R}_+$ satisfying $\sum_{i=1}^m w_i > 0$, the weighted Fréchet mean M over the metric space $\{\mathcal{S}_{++}^n, d^{\text{ALE}}\}$ has a closed form

$$M = \log_\alpha^{-1} \left(\sum_{i=1}^m \frac{w_i}{\sum_{j=1}^m w_j} \log_\alpha(S_i) \right). \quad (6.24)$$

Like LEM, although ALEM is not affine-invariant, it enjoys several other invariance properties.

Proposition 157 (Bi-invariance). [↓] ALEM is a Lie group bi-invariant metric.

Proposition 158 (Exponential Invariance). [↓] The Fréchet means under ALEM are exponential-invariant. In other words, for $S_1, \dots, S_m \in \mathcal{S}_{++}^n$ and $\beta \in \mathbb{R}$,

$$(\text{FM}(S_1, \dots, S_m))^\beta = \text{FM}(S_1^\beta, \dots, S_m^\beta), \quad (6.25)$$

where $\text{FM}(S_1, \dots, S_m)$ denotes the Fréchet mean of $S_1, \dots, S_m$.

In addition to exponential invariance, the Fréchet mean induced by our ALEM also satisfies various properties presented in Ando et al. [5].

Proposition 159. [↓] Let the following be SPD matrices:

$$A, B, C, A_0, B_0, C_0 \in \mathcal{S}_{++}^n. \quad (6.26)$$

Let $\text{FM}(A, B, C)$ denote the Fréchet mean of A, B, C under ALEM. Then the Fréchet mean satisfies the following properties.

(1) U1: permutation invariance. For any permutation π of $\{A, B, C\}$,

$$\text{FM}(\pi(A, B, C)) = \text{FM}(A, B, C). \quad (6.27)$$

(2) U2. $\text{FM}(A, A, A) = A$.

The following properties hold if A, B, C, A_0, B_0, C_0 commute.

(1) V1: joint homogeneity.

$$\text{FM}(aA, bB, cC) = (abc)^{1/3} \text{FM}(A, B, C), \quad \forall a, b, c > 0. \quad (6.28)$$

(2) V2: monotonicity. The map $(A, B, C) \mapsto \text{FM}(A, B, C)$ is monotone. Specifically, if $A \geq A_0$, $B \geq B_0$, and $C \geq C_0$, then $\text{FM}(A, B, C) \geq \text{FM}(A_0, B_0, C_0)$ in the positive semidefinite ordering.

(3) V3: self-duality. $\text{FM}(A, B, C) = \text{FM}(A^{-1}, B^{-1}, C^{-1})^{-1}$.

(4) V4: determinant identity. $\det \text{FM}(A, B, C) = (\det A \cdot \det B \cdot \det C)^{1/3}$.

In fact, Thm. 159 holds true for any finite number of SPD matrices. Besides, the geodesic distance induced by ALEMs has similarity invariance.

Proposition 160 (Similarity Invariance). $\llbracket \Downarrow \rrbracket$ The geodesic distance under ALEM is similarity invariant. In other words, let $R \in \text{SO}(n)$ be a rotation matrix and let $s \in \mathbb{R}_{++}$ be a scale factor. Given any two SPD matrices S_1 and S_2 , we have

$$d^{\text{ALE}}(S_1, S_2) = d^{\text{ALE}}(s^2 R S_1 R^\top, s^2 R S_2 R^\top). \quad (6.29)$$

Let us explain a bit more about the above three kinds of invariance. First, among metrics on Lie groups, bi-invariant metrics are the most convenient ones [187, Ch. V]. Second, exponential invariance offers a fast computation for Fréchet means under exponential scaling. Finally, similarity invariance is significant for describing frequently encountered covariance matrices [9].

The above discussion focuses on the theoretical perspective. Now, let us reconsider Eq. (6.12) in a numerical way.

Proposition 161. $\Downarrow$ $\log_\alpha$ can be rewritten as

$$\log_\alpha(S) = U \log_\alpha(\Sigma) U^\top, \quad (6.30)$$

$$= U A \log(\Sigma) U^\top, \quad (6.31)$$

$$= U \frac{\log(\Sigma)}{B} U^\top, \quad (6.32)$$

where $\frac{X}{Y}$ is the diagonal division, $B = \text{diag}(\log(a_1), \dots, \log(a_n))$, and $A = \frac{I_n}{B}$.

Based on the above proposition, more analyses could be carried out from a numerical point of view. First, $\log_\alpha(\cdot)$ can balance the eigenvalues of an input SPD matrix S by exploiting different bases for different eigenvalues. In Riemannian algorithms, manifold-valued features usually contain vibrant information. We expect that by the above adaptation, manifold-valued data could be better fitted and the learning ability of algorithms could be further promoted.

Remark 162. Note that the discussion in Sec. 6.2.2.3 and Sec. 6.2.2.5 can also be readily transferred to LCM, generating an adaptive version of LCM.

6.2.3 Parameter Learning

As shown in Thm. 152, Riemannian computations under ALEM are built upon the general matrix logarithm $\log_\alpha$ and its inverse. Accordingly, this subsection studies the backpropagation and parameter optimization of these two maps.

6.2.3.1 Gradient Computation

For both the general matrix logarithm and exponential, gradients are required with respect to their parameters and inputs. Since $\log_\alpha$ involves a structured matrix decomposition, the following derivations rely heavily on structured-matrix backpropagation (BP) [110], whose key idea is invariance of the first-order differential form. The general matrix logarithm is a special case of eigenvalue functions. Based on the formula given by Bhatia [20] and the matrix BP techniques presented by Ionescu et al. [110], we can obtain all the gradients in the following propositions.

Proposition 163. $\Downarrow$ Let us denote $X = \log_\alpha(S)$, where $S \in \mathcal{S}_{++}^d$ is the input SPD matrix. The input gradient $\nabla_S L$ is obtained by specializing the Daleckiĭ–Krein expression in Eqs. (2.90) and (2.91) with $V = \nabla_X L$ and $f(\sigma_i) = A_{ii} \log(\sigma_i)$.

6.2. Adaptive Log-Euclidean Metrics

Name	Detail	Constraint	Method
RELU	Optimizing base vector α (Eq. (6.30))	Positive	shift-ReLU $\max(\epsilon, \alpha)$
MUL	Optimizing diagonal elements of A (Eq. (6.31))	Unconstrained	Standard BP
DIV	Optimizing diagonal elements of B (Eq. (6.32))	Unconstrained	Standard BP

Table 6.1: Parameter learning for the general matrix logarithm and exponential.

The parameter gradient is

$$\nabla_A L = [U^\top (\nabla_X L) U] \circledast \log(\Sigma), \quad (6.33)$$

where $S = U\Sigma U^\top$ is the eigendecomposition of an SPD matrix and $\sigma_1, \dots, \sigma_d$ are the diagonal entries of Σ .

As the inverse map corresponding to Eq. (6.31), the general matrix exponential $\log_\alpha^{-1}$ can be rewritten as

$$\begin{aligned} \log_\alpha^{-1}(X) &= U \operatorname{diag}(a_1^{\Sigma_{11}}, \dots, a_n^{\Sigma_{nn}}) U^\top \\ &= U \exp\left(\frac{\Sigma}{A}\right) U^\top, \end{aligned} \quad (6.34)$$

where $X = U\Sigma U^\top \in \mathcal{S}^n$ is an eigendecomposition of X . Following Thm. 163, we obtain the backpropagation of $\log_\alpha^{-1}$.

Proposition 164. [↓] Let us denote $X = \log_\alpha^{-1}(S)$ with $S \in \mathcal{S}^d$. The input gradient $\nabla_S L$ is obtained by specializing the Daleckiĭ–Krein expression in Eqs. (2.90) and (2.91) with $V = \nabla_X L$ and $f(\sigma_i) = e^{\frac{\sigma_i}{A_{ii}}}$. The parameter gradient is

$$\nabla_A L = [U^\top (\nabla_X L) U] \circledast \left(\operatorname{diag}\left(a_1^{\Sigma_{11}}, \dots, a_d^{\Sigma_{dd}}\right) \frac{-\Sigma}{A^2} \right), \quad (6.35)$$

where $S = U\Sigma U^\top$ is the eigendecomposition of a symmetric matrix and $\sigma_1, \dots, \sigma_d$ are the diagonal entries of Σ .

6.2.3.2 Parameter Updates

The general matrix logarithm and exponential share the same base parameters. Let us focus on the former. Let the input SPD matrix S have dimension $d \times d$. Recalling Eqs. (6.30) to (6.32), there are three ways to implement parameter learning. We could learn the base vector α in Eq. (6.30), diagonal matrix A in Eq. (6.31), or diagonal

matrix B in Eq. (6.32), respectively.

For learning A in Eq. (6.31) or B in Eq. (6.32), since the parameters (diagonal elements) lie in a Euclidean space $\mathbb{R}^d$, the optimization can be easily integrated into the BP algorithm. We call learning A MUL and learning B DIV.

For learning α in Eq. (6.30), each element a of α satisfies $a > 0$ and $a \neq 1$. Since the equality case can be avoided by setting $a = 1 + \epsilon$ whenever $a = 1$, where $\epsilon \in \mathbb{R}_{++}$, it remains to enforce positivity during optimization. We consider two strategies for doing so. The first strategy applies the shift-ReLU $\max(\epsilon, a)$ to an unconstrained parameter. We call this strategy RELU. Other transformations, such as squaring the parameter, are also feasible, but we focus on RELU. The second strategy takes a geometric approach by viewing a as a point on a one-dimensional SPD manifold and optimizing it using the Riemannian optimization strategy reviewed in Sec. 2.7. We call this strategy GEOM. Its RSGD update is given in the following proposition.

Proposition 165. $\Downarrow$ *Viewing a positive scalar a as a point in a one-dimensional SPD manifold, we have the following RSGD update formula.*

$$a^{(t+1)} = a^{(t)} e^{-\gamma^{(t)} a^{(t)} \nabla_{a^{(t)}} L}, \quad (6.36)$$

where $\nabla_{a^{(t)}} L$ is the Euclidean gradient of L with respect to a at $a^{(t)}$, $\gamma^{(t)}$ is the learning rate, and $e^{(\cdot)}$ is the natural exponential function.

Moreover, the following proposition shows that GEOM is equivalent to DIV.

Proposition 166. $\Downarrow$ *For parameter learning in $\log_\alpha$, optimizing the base vector α by RSGD is equivalent to optimizing the divisor matrix B by Euclidean stochastic gradient descent (ESGD).*

Consequently, the three distinct update schemes are RELU, DIV, and MUL, as summarized in Tab. 6.1.

6.2.4 Experiments

In this section, we validate the efficacy of our approaches on multiple data sets. Riemannian metrics are foundational to Riemannian neural networks. Therefore, our ALEM can redesign basic blocks in Riemannian neural networks. Beyond the main SPDNet experiments, we apply our ALEM to other Riemannian building blocks, including the LieBN framework in Chapter 3, Riemannian residual blocks [117], and Riemannian classifiers [159]. More details on data sets and experimental settings are provided in

6.2. Adaptive Log-Euclidean Metrics

Learning rate		$1e^{-2}$			$5e^{-2}$	
Architecture	{ 93, 30}	{ 93, 70, 30}	{ 93, 70, 50, 30}	{ 93, 30}	{ 93, 70, 30}	{ 93, 70, 50, 30}
SPDNet	62.92±0.81	62.87±0.60	63.03±0.67	63.89±0.73	64.00±0.65	63.72±0.61
SPDNetBN	63.03±0.75	58.27±1.7	52.02±2.34	63.75±0.69	48.78±5.15	37.84±6.10
ALog-MUL	63.52±0.75	63.86±0.58	63.94±0.44	64.4±0.68	64.60±0.69	64.36±0.49
ALog-DIV	63.60±0.79	63.93±0.52	63.81±0.7	64.81±0.64	64.84±0.65	64.80±0.36
ALog-RELU	63.02±0.79	63.94±0.64	63.14±0.65	63.97±0.75	64.10±0.63	63.78±0.46

Table 6.2: Results of ALog on the HDM05 data set. The best results are **bold**.

Secs. [A.1](#) and [A.3.7](#).

6.2.4.1 Applications in SPDNet

In existing SPD neural networks, activation and classification layers commonly map SPD features into the logarithmic domain through the matrix logarithm [106, 232, 37, 156, 47]. This mapping is an isomorphism that identifies the SPD manifold under LEM with the Euclidean space $\mathcal{S}^n$. Replacing the natural matrix logarithm $\log$ with the learnable general logarithm $\log_\alpha$ allows the resulting layer to adapt the underlying geometry to the learned SPD features.

We focus on the classic SPDNet [106], whose BiMap, ReEig, and LogEig layers are reviewed in Sec. [A.2.1](#). Specifically, replacing the matrix logarithm in its LogEig layer with the learnable general matrix logarithm $\log_\alpha$ yields the *Adaptive Logarithm (ALog)* layer. We compare SPDNet, SPDNetBN, and the ALog-RELU/MUL/DIV variants on HDM05, FPFA, and AFEW.

On the three data sets, the numbers of training epochs are set to 200, 500, and 100. We verify our ALog on SPDNet with various architectures. In addition, we test the robustness of the proposed layer against different learning rates on the HDM05 and FPFA data sets. Generally speaking, among all three implementations, **ALog-MUL** shows the most robust performance gain and achieves consistent improvement over the vanilla matrix logarithm. We also observe that ALog-MUL is comparable to or even better than SPDNetBN, which, however, introduces substantially greater complexity than our approach. The main reason for the superiority of our ALog against the vanilla matrix logarithm is that our ALog can adaptively respect the vibrant geometry of SPD manifolds, depending on the characteristics of data sets, while only LEM can be respected by the matrix logarithm. The following are detailed observations and analyses.

Results on the HDM05 data set. The 10-fold results are presented in Tab. [6.2](#), where the data split and weight initialization are randomized. Following Huang and

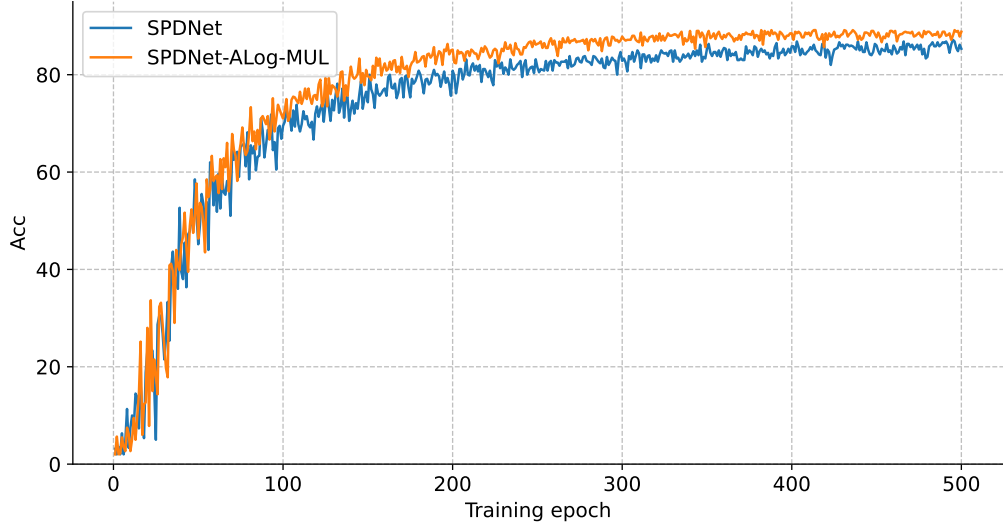


Figure 6.1: Accuracy curves on the FPHA data set.

SPDNet	SPDNetBN	ALog		
		MUL	DIV	RELU
85.73 $\pm$ 0.80	86.83 $\pm$ 0.74	87.8 $\pm$ 0.71	88.07$\pm$1.13	86.65 $\pm$ 0.68

Table 6.3: Results of ALog on the FPHA data set.

Van Gool [106], three architectures are implemented on this data set, *i.e.*, $\{93, 30\}$, $\{93, 70, 30\}$, and $\{93, 70, 50, 30\}$. Generally speaking, endowed with ALog, SPDNet achieves consistent improvement. Among all three implementations, RELU only brings limited improvement. The reason might be that RELU fails to respect the innate geometry of the positive constraint. There is another interesting observation worth mentioning. In Brooks et al. [31], only the result of SPDNetBN under the architecture of $\{93, 30\}$ is reported on this data set. Our experiments show that with the network going deeper, SPDNetBN tends to collapse, while our ALog layer performs robustly in all settings.

Results on the FPHA data set. We validate our approach on this data set, with a learning rate of $1e^{-2}$, over 10-fold cross-validation on random initialization. Since our experiments indicate that the vanilla SPDNet is already saturated with 1 BiMap layer, we just report the results on the architecture of $\{63, 33\}$, which are presented in Tab. 6.3. Although DIV performs best on this data set, it presents the largest variance. There is an underlying nonlinear scaling mechanism in the update of DIV, which might undermine its robustness. Without loss of generality, let us focus on a single scalar parameter b in Eq. (6.32). The ultimate factor multiplied by the plain logarithm is $1/b$.

Therefore, the change of the multiplier after the update would be

$$1/(b - \Delta) - 1/b = \Delta/[(b - \Delta)b]. \quad (6.37)$$

Eq. (6.37) will scale the original Δ to some extent. This scaling mechanism might undermine the robustness of the ALog layer. However, ALog-MUL achieves robust improvement and even surpasses SPDNetBN. This again demonstrates the significance of our adaptive mechanism for Riemannian deep networks. Finally, in terms of convergence analysis, accuracy curves with and without ALog are also reported in Fig. 6.1.

Results on the AFEW

data set. On this data set, the learning rate is $5e^{-2}$, and we validate our method under four network architectures, *i.e.*, $\{512, 100\}$, $\{512, 200, 100\}$, $\{512, 400, 200, 100\}$, and $\{512, 400, 300, 200, 100\}$. Note that, on this data set, SPDNetBN tends to

Depth	1	2	3	4
SPDNet	48.53	46.89	48.24	47.22
SPDNetBN	46.89	46.65	47.62	48.35
ALog-MUL	48.57	48.13	49.45	50.62
ALog-DIV	48.42	48.02	48.13	49.89
ALog-RELU	48.06	47.25	48.86	48.1

Table 6.4: Results of ALog on the AFEW data set.

present relatively large fluctuations in performance, so we compute the median of the last ten epochs. On various architectures, consistent improvement can be observed when SPDNet is endowed with our ALog. In addition, MUL performs best among all three implementations. Another interesting observation is that SPDNetBN seems ineffective on these deep features, while our methods show consistently superior performance, most notably for ALog-MUL. This indicates that our adaptive layer maintains effectiveness when applied to covariance matrices from deep features.

Model complexity. Our ALog manifests the same complexity, no matter how it is optimized. Without loss of generality, the discussion below focuses on ALog-MUL. The extra computation and memory costs caused by the ALog layer are minor. It only depends on the final dimension of the network. Let us take the deepest one on the AFEW data set as an example. Our ALog only brings 100 unconstrained scalar parameters, while SPDNetBN needs an SPD matrix parameter for each RBN layer. The total number of parameters in RBN layers is $400^2 + 300^2 + 200^2$, which is much larger than ours. In addition, SPDNetBN needs to store the running mean of SPD matrices in every RBN layer, while our ALog only needs to store a vector. In terms of computation, the extra cost of our ALog is secondary as well. The forward and backward computation of our ALog is generally the same as the plain matrix logarithm, while computation in

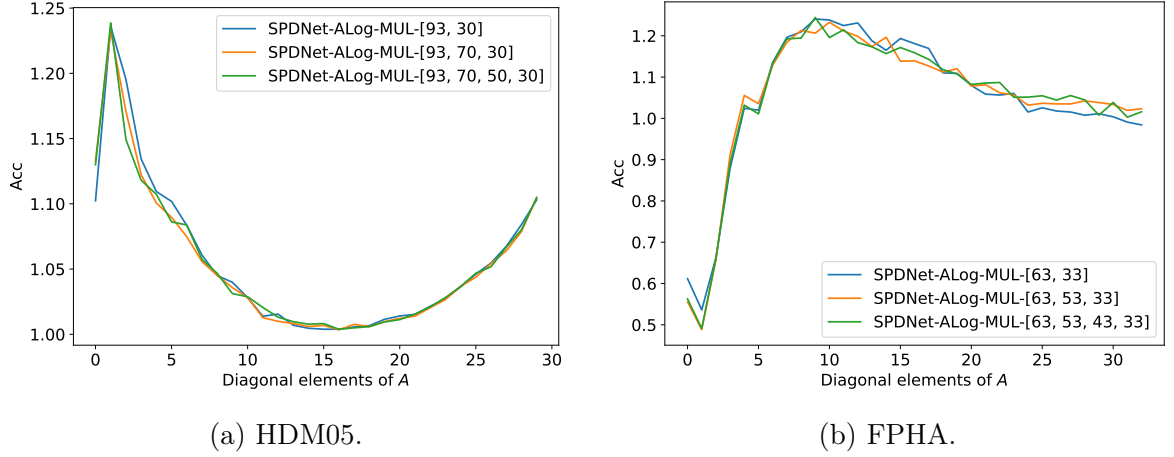


Figure 6.2: Visualization of parameters in the ALog layer on the HDM05 and FPHA data sets.

Data Set Architecture	HDM05			FPHA
	{93, 30}	{93, 70, 30}	{93, 70, 50, 30}	{63, 33}
SPDNet-Log2	63.93±0.81	63.54±0.50	63.98±0.63	86.65±0.67
SPDNet	63.89±0.73	64.00±0.65	63.72±0.61	85.73±0.80
SPDNet-Log10	63.45±0.33	63.8±0.71	63.64±0.64	78.42±0.77
SPDNet-ALog-MUL	64.4±0.68	64.60±0.69	64.36±0.49	87.8±0.71

Table 6.5: Results of fixed bases on the HDM05 and FPHA data sets.

the RBN layer is much more complex. All in all, our ALog can consistently improve the performance of SPDNet and achieve comparable or better results than SPDNetBN with much lower computation and memory costs.

Visualization. We visualize the final learned parameters of the ALog layer. Since ALog-MUL is the most robust strategy, we visualize the parameters of ALog-MUL. Specifically, we plot the final values of the diagonal elements of A in Eq. (6.31) and visualize the results in Figs. 6.2a and 6.2b. We observe that the distribution of the parameters is consistent within the same data set but varies between data sets. This indicates that our approach can capture vibrant patterns in different data sets, respecting their specific geometry.

Ablation studies. To further demonstrate the utility of the adaptive mechanisms in our approach, we validate the ALog layer with fixed bases. As decimal and binary are the two most common systems, we use $\log_{10}$ and $\log_2$ as examples of shrinking and expanding the natural logarithm $\log$. Specifically, we set the scalar base α in $\log_\alpha$ to 10 and 2 in Eq. (6.30), respectively. We refer to the network with binary/decimal base as SPDNet-Log2/SPDNet-Log10. Note that when $\alpha = e$, $\log_\alpha = \log$, and Eq. (6.30)

Method	Geometry	[93, 30]	[93, 70, 30]	[93, 70, 50, 30]
None	N/A	63.89±0.73	64.00±0.65	63.72±0.61
SPDNetBN	AIM	63.75±0.69	48.78±5.15	37.84±6.10
SPDBN	AIM	64.33±0.89	64.31±0.92	63.62±1.21
LieBN-LEM	LEM	63.67±0.85	65.77±0.89	65.34±0.83
LieBN-ALEM	ALEM	65.24±0.71	70.11±0.96	68.86±0.72

Table 6.6: Comparison of RBN methods on the HDM05 data set.

reduces to the vanilla matrix logarithm. The network is then our baseline, *i.e.*, SPDNet. We conduct 10-fold experiments on the HDM05 and FPHA data sets and set the learning rate to $5e^{-2}$ and $1e^{-2}$, respectively, while keeping the other settings consistent with previous experiments. The results are presented in Tab. 6.5. We observe that the fixed logarithms show similar or slightly worse results than the vanilla log, while our ALog shows consistent improvement. Besides, $\log_{10}$ does not converge on the FPHA data set. In fact, $\log_{10}$ could shrink the gradient, slowing down convergence, especially under a small learning rate. In contrast, our ALog maintains consistent effectiveness. In summary, our ALog can respect vibrant geometry induced by $\log_{\alpha}$ and thus benefit SPD network learning.

6.2.4.2 Riemannian Batch Normalization

The LieBN framework, its normalization guarantee, and its SPD manifestations are developed in Chapter 3. As shown in Thm. 152, $\{\mathcal{S}_{++}^n, \oplus^{\text{ALE}}\}$ forms a Lie group. Besides, Thm. 157 demonstrates that ALEM is bi-invariant with respect to this group structure. Therefore, LieBN under ALEM can also normalize Riemannian sample statistics. Following the LieBN algorithm and its SPD specialization, we implement LieBN under ALEM, denoted LieBN-ALEM. We compare it against AIM-based SPDNetBN [31] and SPDBN [124], and against LieBN under LEM.

Following previous work [124, 31], we adopt the SPDNet backbone. Tab. 6.6 presents the 10-fold average results on the HDM05 data set under different network architectures. Our LieBN-ALEM achieves the best performance among the RBN methods. In particular, the AIM-based SPDNetBN brings worse performance under deeper architectures. In contrast, our LieBN-ALEM can consistently improve the performance across different architectures. Besides, compared with LieBN-LEM, our LieBN-ALEM shows better performance, demonstrating the effectiveness of our ALEM.

6.2.4.3 Riemannian Residual Blocks

The general RResNet construction and SPD residual block are reviewed in Sec. A.2.5. Since its Riemannian exponential is metric-dependent, the Riemannian residual block under ALEM is obtained by substituting Eq. (6.17) into that construction. The required backpropagation of $\log_\alpha^{-1}$ is derived in Thm. 164.

Following Katsman et al. [117], we compare RResNet under different geometries on the HDM05 and NTU60 data sets. Tab. 6.7 reports the 10-fold and 5-fold average results on these data sets. Compared with the vanilla SPDNet, RResNet-AIM brings little improvement, while LEM and ALEM show much better performance. In particular, the ALEM-based RResNet can bring a clear performance improvement, underscoring the effectiveness of our ALEM.

Method	HDM05	NTU60
SPDNet	63.89±0.73	45.90±1.11
RResNet-AIM	63.82±0.58	45.22 ± 1.23
RResNet-LEM	66.51±0.93	48.73±0.60
RResNet-ALEM	69.03±1.06	57.09±0.59

Table 6.7: Experiments on RResNet under different geometries.

6.2.4.4 Riemannian Classifiers

Euclidean MLR, which consists of FC and softmax, has become a standard classification block in Euclidean neural networks. Inspired by this, Nguyen and Yang [159] extended Euclidean MLR to the SPD manifold using gyrostructures [157] for intrinsic classification, referred to as gyro MLR. Three gyro MLRs under LCM,

Learning rate	$1e^{-2}$	$5e^{-2}$
GyroMLR-AIM	54.28±0.47	41.41±0.71
GyroMLR-LCM	42.68±0.88	42.06±0.49
GyroMLR-LEM	53.22±0.47	39.62±1.30
GyroMLR-ALEM	56.21±0.39	51.65±0.44

Table 6.8: Comparison of Gyro MLRs on the NTU60 data set.

AIM, and LEM were introduced by Nguyen and Yang [159]. Following the logic in Nguyen and Yang [159, Sec. 2.4.2], we can obtain the gyro MLR under ALEM.

Theorem 167 (Gyro MLR). [↓] *Given an SPD feature $S \in \mathcal{S}_{++}^n$ and C classes, the SPD gyro MLR under ALEM computes the multinomial probability of each class:*

$$p(y = k \mid S) \propto \exp \left[\left\langle \log_\alpha(S) - \log_\alpha(P_k), (\log_\alpha)_{*, P_k}(\tilde{A}_k) \right\rangle \right], \quad (6.38)$$

where $k \in \{1, \dots, C\}$, $P_k \in \mathcal{S}_{++}^n$, and $\tilde{A}_k \in T_{P_k} \mathcal{S}_{++}^n$.

Following Chapter 4, we set $\tilde{A}_k = \text{PT}_{I_n \rightarrow P_k}(A_k)$ with $A_k \in T_{I_n} \mathcal{S}_{++}^n$. Therefore, the RHS of Eq. (6.38) becomes

$$\exp \left[\left\langle \log_\alpha(S) - \log_\alpha(P_k), (\log_\alpha)_{*, I_n}(A_k) \right\rangle \right]. \quad (6.39)$$

As $(\log_\alpha)_{*, I_n}(A_k) \in T_0 \mathcal{S}^n \cong \mathcal{S}^n$, we view $(\log_\alpha)_{*, I_n}(A_k)$ as the parameter.

We use SPDNet as the backbone. We compare Gyro MLR under our ALEM with those under LEM, LCM, and AIM on the NTU60 data set. Tab. 6.8 presents the 5-fold average results under different learning rates. Our ALEM outperforms the other metrics within the gyro MLR framework. When the learning rate is $5e^{-2}$, our GyroMLR-ALEM shows a larger performance advantage, especially compared with GyroMLR-LEM. These results demonstrate that Riemannian networks can benefit from the adaptivity of our ALEM.

6.3 Product Cholesky Metrics

6.3.1 Introduction

Whereas Sec. 6.2 introduces adaptive flexibility through pullback Euclidean geometry, this part pursues a complementary route centered on computational efficiency and numerical stability. LCM provides the natural bridge between these routes. It combines simple closed-form Riemannian operators with the fast and stable computation of the Cholesky decomposition [137]. LCM is induced by the Cholesky decomposition from a Riemannian metric on the Cholesky manifold, which is the space of lower triangular matrices with positive diagonal entries. We refer to this source metric as the diagonal log metric. Its interpretation as a pullback Euclidean metric was established in Thm. 147. We reveal a simple *product structure* underlying the diagonal log metric: a Euclidean metric on the strictly lower triangular part together with n copies of a Riemannian metric on $\mathbb{R}_{++}$ for the diagonal part. This observation opens up a principled design space, as any metric on $\mathbb{R}_{++}$ can induce a metric on the Cholesky manifold, and further yield a corresponding metric on the SPD manifold via the Cholesky decomposition.

Building on this product structure, we introduce two Cholesky metrics, the diagonal power metric and the diagonal Bures–Wasserstein metric, which induce two SPD metrics via the Cholesky decomposition: the *Power-Cholesky Metric (PCM)* and

Bures–Wasserstein–Cholesky Metric (BWCM). Unlike the diagonal logarithm in LCM, our metrics rely on diagonal powers, improving numerical stability by avoiding exponentiation and logarithms. We further define in the Cholesky factors of SPD matrices a diagonal power deformation, which continuously connects existing and new metrics. As power $\theta \rightarrow 0$, the deformed metric converges to LCM, while at $\theta = 1$ it recovers our proposed metrics, thereby offering a tunable trade-off. All proposed SPD metrics admit closed-form Riemannian operators, including geodesics, logarithmic and exponential maps, parallel transport, as well as gyrovector operators [200], which extend vector addition and scalar multiplication into manifolds. These operators make our metrics directly applicable to SPD neural networks. In particular, by substituting these operators into the Riemannian MLR formulation developed in Chapter 4 and residual blocks [118], we directly obtain SPD MLR classifiers and residual blocks under our metrics. We validate our metrics with experiments on SPD neural networks, numerical stability analyses, and tensor interpolation, showing the effectiveness, efficiency, and robustness of our metrics. In summary, our main **contributions** are:

- (1) Revealing the underlying simple product structure in the Cholesky manifold;
- (2) Proposing two Cholesky metrics and their SPD counterparts, PCM and BWCM, which admit fast and stable closed-form operators;
- (3) Developing SPD classifiers and residual blocks based on our metrics for SPD neural networks.³

Outline. Sec. 6.3.2 recalls the Cholesky geometry used in this part. Sec. 6.3.3 develops the Cholesky product geometries, and Sec. 6.3.4 derives their SPD counterparts. Their applications to SPD neural networks and the experimental evaluation are presented in Secs. 6.3.5 and 6.3.6. Proofs are deferred to Sec. B.10.

6.3.2 Preliminaries

Pullback metrics are defined in Thm. 33, while the commonly used SPD geometries are summarized in Tabs. 2.5 and 2.6. We further recall the Cholesky geometry. The Euclidean space of $n \times n$ lower triangular matrices is denoted LT^n . Its open subset, whose diagonal elements are all positive, is denoted by $\mathcal{L}_{++}^n$. The Cholesky space $\mathcal{L}_{++}^n$ forms a submanifold of LT^n [137]. For a Cholesky matrix $L \in \mathcal{L}_{++}^n$ and tangent vectors $X, Y \in T_L \mathcal{L}_{++}^n$, the Riemannian metric on the Cholesky manifold, referred to as the

³The code is available at https://github.com/GitZH-Chen/PCM_BWCM.

diagonal log metric, is

$$g_L^{\text{DL}}(X, Y) = \langle [X], [Y] \rangle + \langle \mathbb{L}^{-1}\mathbb{X}, \mathbb{L}^{-1}\mathbb{Y} \rangle. \quad (6.40)$$

Here, $[X]$ and $[Y]$ are the strictly lower triangular parts of X and Y , while $\mathbb{L}$, $\mathbb{X}$, and $\mathbb{Y}$ are the diagonal matrices formed from their diagonal entries. LCM is the pullback metric of g^{DL} by the Cholesky decomposition. As shown in Thm. 147, the diagonal log metric is the pullback metric, by the diagonal log map, of the Euclidean metric over LT^n , which rationalizes our nomenclature.

6.3.3 Product Geometries on the Cholesky

We first unveil the product structure beneath the existing diagonal log metric on the Cholesky manifold. Based on this, we propose two novel Cholesky metrics.

6.3.3.1 Disentangling the Cholesky Geometry

We denote the space of $n \times n$ diagonal matrices with positive diagonal elements by $\text{Diag}^+(n)$ and the space of $n \times n$ strictly lower triangular matrices by $\text{LT}^0(n)$. Then, $\text{LT}^0(n)$ is a Euclidean space, and $\text{Diag}^+(n) \cong (\mathbb{R}_{++})^n$ is an open submanifold of $\mathbb{R}^n$. Recalling Eq. (6.40), it is defined separately on $\text{LT}^0(n)$ and $\text{Diag}^+(n)$. Besides, $\text{Diag}^+(n)$ can be identified as the product of n copies of $\mathbb{R}_{++}$. The above discussion implies a product structure. We denote the standard Euclidean metric over $\text{LT}^0(n)$ by g^{E} and define the Riemannian metric on $\mathbb{R}_{++}$ as

$$g_p^{\mathbb{R}_{++}}(v, w) = p^{-2}vw, \quad \forall p \in \mathbb{R}_{++} \text{ and } v, w \in T_p\mathbb{R}_{++}. \quad (6.41)$$

Then, $\mathcal{L}_{++}^n$ is the product manifold of $\text{LT}^0(n)$ and n copies of $\mathbb{R}_{++}$:

$$\{\mathcal{L}_{++}^n, g^{\text{DL}}\} = \{\text{LT}^0(n), g^{\text{E}}\} \times \overbrace{\{\mathbb{R}_{++}, g^{\mathbb{R}_{++}}\} \times \cdots \times \{\mathbb{R}_{++}, g^{\mathbb{R}_{++}}\}}^n. \quad (6.42)$$

6.3.3.2 Product Geometries on the Cholesky

The following definition characterizes the underlying product structure in Eq. (6.42).

Definition 168 (Product geometries). Suppose g^{LT^0} is a Euclidean inner product on $\text{LT}^0(n)$ and $\{g^i\}_{i=1}^n$ are Riemannian metrics on $\mathbb{R}_{++}$. Then, the weighted product

metric g on $\mathcal{L}_{++}^n$ is defined as $g_L(X, Y) = g^{\text{LT}^0}(\lfloor X \rfloor, \lfloor Y \rfloor) + \sum_{i=1}^n \alpha_i g_{L_{ii}}^i(X_{ii}, Y_{ii})$, with $L \in \mathcal{L}_{++}^n$, $X, Y \in T_L \mathcal{L}_{++}^n$, and $\alpha_i > 0$. Here, $\lfloor X \rfloor$ and $\lfloor Y \rfloor$ are the strictly lower triangular parts of X and Y , while L_{ii} , X_{ii} , and Y_{ii} are the i -th diagonal elements.

For simplicity, we focus on the case where $g^{\text{LT}^0} = g^{\text{E}}$ is the standard Euclidean metric, all α_i are equal to 1, and all g^i are identical. Since $\mathbb{R}_{++}$ can be viewed as a one-dimensional SPD manifold $\mathcal{S}_{++}^1$, the Riemannian metrics reviewed in Sec. 2.9.1 can be immediately used to build Riemannian metrics on the Cholesky manifold. We additionally consider the Generalized Bures–Wasserstein Metric (GBWM), which is the pullback of BWM by $S \mapsto M^{-\frac{1}{2}} S M^{-\frac{1}{2}}$ for $S, M \in \mathcal{S}_{++}^n$ [93]. For clarity, we denote PEM and GBWM by θ -EM and M -BWM, respectively. Simple computations show that AIM, LEM, and LCM coincide with Eq. (6.41) on $\mathcal{S}_{++}^1$. Therefore, these metrics reduce to three classes on $\mathcal{S}_{++}^1$: (1) LEM, LCM, or AIM; (2) θ -EM; (3) M -BWM or BWM. When the metric on $\mathbb{R}_{++}$ is AIM (LEM or LCM), the resulting product metric on the Cholesky manifold is the diagonal log metric, and the pullback SPD metric via the Cholesky decomposition is exactly LCM.

Inspired by the above analysis, we obtain two new metrics on the Cholesky manifold by setting each g^i in Thm. 168 to θ -EM and M -BWM, termed the *Diagonal Power Metric* (θ -DPM) and *Diagonal Bures–Wasserstein Metric* ($\mathbb{M}$ -DBWM with $\mathbb{M} \in \text{Diag}^+(n)$), respectively. By product geometries [131], we can obtain closed-form expressions for their Riemannian operators, such as the geodesic, logarithmic map, parallel transport, and weighted Fréchet mean. These operators are particularly important for building concrete learning algorithms [227, 132, 31, 141].

Theorem 169 (θ -DPM). $\Downarrow$ Let $L, K \in \mathcal{L}_{++}^n$ and $X, Y \in T_L \mathcal{L}_{++}^n$, and let $\{L_i \in \mathcal{L}_{++}^n\}_{i=1}^N$ have weights $\{w_i\}_{i=1}^N$ satisfying $w_i > 0$ for all i and $\sum_{i=1}^N w_i = 1$. Then, the Riemannian operators under θ -DPM with $\theta \neq 0$ are

$$g_L^{\theta\text{-DE}}(X, Y) = \langle \lfloor X \rfloor, \lfloor Y \rfloor \rangle + \langle \mathbb{L}^{\theta-1} \mathbb{X}, \mathbb{L}^{\theta-1} \mathbb{Y} \rangle, \quad (6.43)$$

$$\gamma_{(L, X)}(t) = \lfloor L \rfloor + t \lfloor X \rfloor + \mathbb{L} (I_n + t \theta \mathbb{L}^{-1} \mathbb{X})^{\frac{1}{\theta}}, \quad (6.44)$$

$$\text{Log}_L(K) = \lfloor K \rfloor - \lfloor L \rfloor + \frac{1}{\theta} \mathbb{L} \left[(\mathbb{L}^{-1} \mathbb{K})^\theta - I_n \right], \quad (6.45)$$

$$\text{PT}_{L \rightarrow K}(X) = \lfloor X \rfloor + (\mathbb{L}^{-1} \mathbb{K})^{1-\theta} \mathbb{X}, \quad (6.46)$$

$$\text{d}^2(L, K) = \|\lfloor K \rfloor - \lfloor L \rfloor\|_{\text{F}}^2 + \frac{1}{\theta^2} \|\mathbb{K}^\theta - \mathbb{L}^\theta\|_{\text{F}}^2, \quad (6.47)$$

$$\text{WFM}(\{w_i\}, \{L_i\}) = \sum_i w_i \lfloor L_i \rfloor + \left(\sum_i w_i \mathbb{L}_i^\theta \right)^{\frac{1}{\theta}}, \quad (6.48)$$

where $\|\cdot\|_F$ is the Frobenius norm. $\mathbb{X}, \mathbb{Y}, \mathbb{L}, \mathbb{K}$, and $\mathbb{L}_i$ are diagonal matrices with diagonal elements from X, Y, L, K , and L_i . $\gamma_{(L,X)}(t)$ denotes the geodesic starting at L with initial velocity X . $\text{PT}_{L \rightarrow K}(\cdot)$ is the parallel transport along the geodesic connecting L and K . Log , d , and WFM are the Riemannian logarithm, geodesic distance, and weighted Fréchet mean, respectively. Note that $\gamma_{(L,X)}(t)$ is locally defined in $\{t \in \mathbb{R} | \mathbb{L} + t\theta\mathbb{X} \in \text{Diag}^+(n)\}$.

Theorem 170 (M-DBWM). $\Downarrow$ Following the notation in Thm. 169, the Riemannian operators under M-DBWM with $\mathbb{M} \in \text{Diag}^+(n)$ are

$$g_L^{\mathbb{M}\text{-DBW}}(X, Y) = \langle \lfloor X \rfloor, \lfloor Y \rfloor \rangle + \frac{1}{4} \langle \mathbb{L}^{-1} \mathbb{X}, \mathbb{M}^{-1} \mathbb{Y} \rangle, \quad (6.49)$$

$$\gamma_{(L,X)}(t) = \lfloor L \rfloor + t \lfloor X \rfloor + \mathbb{L} \left(I_n + t \frac{1}{2} \mathbb{L}^{-1} \mathbb{X} \right)^2, \quad (6.50)$$

$$\text{Log}_L(K) = \lfloor K \rfloor - \lfloor L \rfloor + 2\mathbb{L} \left[(\mathbb{L}^{-1} \mathbb{K})^{\frac{1}{2}} - I_n \right], \quad (6.51)$$

$$\text{PT}_{L \rightarrow K}(X) = \lfloor X \rfloor + (\mathbb{L}^{-1} \mathbb{K})^{\frac{1}{2}} \mathbb{X}, \quad (6.52)$$

$$d^2(L, K) = \|\lfloor K \rfloor - \lfloor L \rfloor\|_F^2 + \|\mathbb{M}^{-\frac{1}{2}} \left(\mathbb{K}^{\frac{1}{2}} - \mathbb{L}^{\frac{1}{2}} \right)\|_F^2, \quad (6.53)$$

$$\text{WFM}(\{w_i\}, \{L_i\}) = \sum_i w_i \lfloor L_i \rfloor + \left(\sum_i w_i \mathbb{L}_i^{\frac{1}{2}} \right)^2, \quad (6.54)$$

where the geodesic $\gamma_{(L,X)}(t)$ is locally defined in $\{t \in \mathbb{R} | \mathbb{L} + \frac{t}{2} \mathbb{X} \in \text{Diag}^+(n)\}$. When $\mathbb{M} = I_n$ in M-DBWM, the resulting metric is denoted by DBWM.

On the SPD manifold, GBWM is locally AIM [93]. Similarly, on the Cholesky manifold, our M-DBWM is locally the diagonal log metric at $L \in \mathcal{L}_{++}^n$: $g_L^{\mathbb{L}\text{-DBW}}(X, Y) = \langle \lfloor X \rfloor, \lfloor Y \rfloor \rangle + \frac{1}{4} \langle \mathbb{L}^{-1} \mathbb{X}, \mathbb{L}^{-1} \mathbb{Y} \rangle$. GBWM on the SPD manifold generally has no closed-form expression for the Fréchet mean [22]. Moreover, the closed-form expression of parallel transport under BWM is known only if two SPD matrices commute [196]. In contrast, all these operators have closed-form expressions under M-DBWM on $\mathcal{L}_{++}^n$.

6.3.3.3 Deformed Cholesky Metrics

On SPD manifolds, metrics deformed by the matrix power can interpolate between a given metric and an LEM-like metric [194, Sec. 3.1]. Inspired by this, we define

a diagonal power deformation on the Cholesky manifold. For $\theta \neq 0$, we denote the diagonal power by $\text{DPow}_\theta : \text{Diag}^+(n) \ni \mathbb{P} \mapsto \mathbb{P}^\theta \in \text{Diag}^+(n)$. We will show how our proposed metric is connected to the existing diagonal log metric by DPow_θ .

Definition 171. Let $\{\mathcal{L}_{++}^n, g\} = \{\text{LT}^0(n), g^E\} \times \{\text{Diag}^+(n), \tilde{g}\}$ be a product metric and $\theta \neq 0$. We define the diagonal-power-deformed metric of g as $\{\mathcal{L}_{++}^n, g^\theta\} = \{\text{LT}^0(n), g^E\} \times \{\text{Diag}^+(n), \frac{1}{\theta^2} \text{DPow}_\theta^* \tilde{g}\}$.

The following lemma shows that g^θ in Thm. 171 converges to a diagonal-log-like metric as $\theta \rightarrow 0$.

Lemma 172. [↓] Given $L \in \mathcal{L}_{++}^n$ and $X, Y \in T_L \mathcal{L}_{++}^n$, g^θ in Thm. 171 satisfies

$$g_L^\theta(X, Y) = \langle [X], [Y] \rangle + \tilde{g}_{\mathbb{L}^\theta}(\mathbb{L}^{\theta-1} \mathbb{X}, \mathbb{L}^{\theta-1} \mathbb{Y}) \xrightarrow{\theta \rightarrow 0} \langle [X], [Y] \rangle + \tilde{g}_{I_n}(\mathbb{L}^{-1} \mathbb{X}, \mathbb{L}^{-1} \mathbb{Y}). \quad (6.55)$$

Now, we discuss the deformation of the diagonal log metric, θ -DPM, and $\mathbb{M}$ -DBWM. First, Eq. (6.55) indicates that the diagonal-power-deformed metric of the diagonal log metric is itself. Second, θ -DPM is the diagonal-power-deformed metric of the Euclidean metric on the Cholesky manifold. Besides, θ -DPM interpolates between the diagonal log metric ($\theta \rightarrow 0$) and the Euclidean metric ($\theta = 1$). Third, the diagonal-power-deformed metric of $\mathbb{M}$ -DBWM, referred to as $(\theta, \mathbb{M})$ -DBWM, is

$$g_L^{(\theta, \mathbb{M})\text{-DBW}}(X, Y) = \langle [X], [Y] \rangle + \frac{1}{4} \langle \mathbb{L}^{\theta-2} \mathbb{X}, \mathbb{M}^{-1} \mathbb{Y} \rangle. \quad (6.56)$$

When $\mathbb{M} = I_n$, the deformed metric of DBWM, *i.e.*, θ -DBWM, tends to be a scaled diagonal log metric as $\theta \rightarrow 0$:

$$\lim_{\theta \rightarrow 0} g_L^{\theta\text{-DBW}}(X, Y) = \langle [X], [Y] \rangle + \frac{1}{4} \langle \mathbb{L}^{-1} \mathbb{X}, \mathbb{L}^{-1} \mathbb{Y} \rangle. \quad (6.57)$$

As $(\theta, \mathbb{M})$ -DBWM is the pullback metric by diagonal power and scaled by a constant, the Riemannian operators also have closed-form expressions, which are discussed in Sec. A.4.5.1.

6.3.3.4 Algebraic Structures

As reviewed in Sec. 2.6, Eqs. (2.77) and (2.78) define gyroaddition and scalar gyromultiplication on a Riemannian manifold. The gyro operations under the diagonal log metric reduce to vector-space operations, as the metric is induced by the Euclidean

metric over the lower triangular matrices. This subsection studies the gyro-structures over θ -DPM and $(\theta, \mathbb{M})$ -DBWM. Let the identity matrix be the origin and $\mathcal{C}$ be θ -DPM or $(\theta, \mathbb{M})$ -DBWM. We have the following.

Lemma 173 (Gyro-structures). $\llbracket \downarrow \rrbracket$ For $L, K \in \mathcal{L}_{++}^n$ and $t \in \mathbb{R}$, the gyro operations are

$$L \oplus^{\mathcal{C}} K = \lfloor L \rfloor + \lfloor K \rfloor + (\mathbb{L}^\beta + \mathbb{K}^\beta - I_n)^{\frac{1}{\beta}}, \quad (6.58)$$

$$t \odot^{\mathcal{C}} L = t \lfloor L \rfloor + (t \mathbb{L}^\beta + (1-t)I_n)^{\frac{1}{\beta}}, \quad (6.59)$$

where $\beta = \theta$ for θ -DPM, and $\beta = \theta/2$ for $(\theta, \mathbb{M})$ -DBWM. $\oplus^{\mathcal{C}}$ requires L and K to satisfy $\mathbb{L}^\beta + \mathbb{K}^\beta - I_n \in \text{Diag}^+(n)$, while $\odot^{\mathcal{C}}$ requires $(1-t)I_n + t\mathbb{L}^\beta \in \text{Diag}^+(n)$.

These gyro operations are defined only under the assumptions in Thm. 173, which arise from the locally defined Riemannian exponential map. Throughout, we impose these assumptions implicitly.

Theorem 174. $\llbracket \downarrow \rrbracket$ When the gyro operations are well defined under the conditions in Thm. 173, $\{\mathcal{L}_{++}^n, \oplus^{\mathcal{C}}\}$ satisfies all the axioms of gyrocommutative gyrogroups (Thms. 51 and 52), and $\{\mathcal{L}_{++}^n, \oplus^{\mathcal{C}}, \odot^{\mathcal{C}}\}$ satisfies all the axioms of gyrovector spaces (Thm. 53).

Corollary 175. The identity element of $\{\mathcal{L}_{++}^n, \oplus^{\mathcal{C}}\}$ is the identity matrix, i.e., $\forall L \in \mathcal{L}_{++}^n, I_n \oplus^{\mathcal{C}} L = L$. The inverses are $\ominus^{\mathcal{C}} L = -1 \odot^{\mathcal{C}} L = -\lfloor L \rfloor + (2I_n - \mathbb{L}^\beta)^{\frac{1}{\beta}}$, for $L \in \{L \in \mathcal{L}_{++}^n \mid 2I_n - \mathbb{L}^\beta \in \text{Diag}^+(n)\}$.

Remark 176. The gyrostructures on the Grassmannian have shown success in building Riemannian algorithms [157, 159]. Like our gyrostructure, the gyrostructures of the Grassmannian also require some assumptions for well-definedness [157, Sec. 3.2]. We therefore examine the well-definedness of the gyrostructures under our metrics. In practice, such positivity constraints can be remedied by numerical techniques. Taking $2I_n - \mathbb{L}^\beta \in \text{Diag}^+(n)$ as an example, one can use $d_i \leftarrow \max(d_i, \varepsilon)$ for each diagonal element d_i with a small constant $\varepsilon > 0$.

6.3.3.5 Numerical Advantages over Diagonal Log Metric

Tab. 6.9 summarizes all the Riemannian and gyro operators. The Riemannian operators under $(\theta, \mathbb{M})$ -DBWM and θ -DPM are mostly computed using the diagonal power

Operators	Diagonal Log Metric	θ -DPM	$(\theta, \mathbb{M})$ -DBWM
$g_L(X, Y)$	$\langle [X], [Y] \rangle + \langle \mathbb{L}^{-1}\mathbb{X}, \mathbb{L}^{-1}\mathbb{Y} \rangle$	$\langle [X], [Y] \rangle + \langle \mathbb{L}^{\theta-1}\mathbb{X}, \mathbb{L}^{\theta-1}\mathbb{Y} \rangle$	$\langle [X], [Y] \rangle + \frac{1}{4} \langle \mathbb{L}^{\theta-2}\mathbb{X}, \mathbb{M}^{-1}\mathbb{Y} \rangle$
$\gamma_{(L, X)}(t)$	$[L] + t[X] + \mathbb{L} \exp(t\mathbb{L}^{-1}\mathbb{X})$	$[L] + t[X] + \mathbb{L} (I_n + t\theta\mathbb{L}^{-1}\mathbb{X})^{\frac{1}{\theta}}$	$[L] + t[X] + \mathbb{L} (I_n + t\frac{\theta}{2}\mathbb{L}^{-1}\mathbb{X})^{\frac{2}{\theta}}$
$\text{Log}_L(K)$	$[K] - [L] + \mathbb{L} \log(\mathbb{L}^{-1}\mathbb{K})$	$[K] - [L] + \frac{1}{\theta}\mathbb{L} \left[(\mathbb{L}^{-1}\mathbb{K})^{\theta} - I_n \right]$	$[K] - [L] + \frac{2}{\theta}\mathbb{L} \left[(\mathbb{L}^{-1}\mathbb{K})^{\frac{\theta}{2}} - I_n \right]$
$\text{PT}_{L \rightarrow K}(X)$	$[X] + (\mathbb{L}^{-1}\mathbb{K})\mathbb{X}$	$[X] + (\mathbb{L}^{-1}\mathbb{K})^{1-\theta}\mathbb{X}$	$[X] + (\mathbb{L}^{-1}\mathbb{K})^{1-\frac{\theta}{2}}\mathbb{X}$
$d^2(L, K)$	$\ [K] - [L]\ _{\mathbb{F}}^2 + \ \log(\mathbb{K}) - \log(\mathbb{L})\ _{\mathbb{F}}^2$	$\ [K] - [L]\ _{\mathbb{F}}^2 + \frac{1}{\theta^2}\ \mathbb{K}^{\theta} - \mathbb{L}^{\theta}\ _{\mathbb{F}}^2$	$\ [K] - [L]\ _{\mathbb{F}}^2 + \frac{1}{\theta^2}\ \mathbb{M}^{-\frac{1}{2}} \left(\mathbb{K}^{\frac{\theta}{2}} - \mathbb{L}^{\frac{\theta}{2}} \right)\ _{\mathbb{F}}^2$
$\text{WFM}(\{w_i\}, \{L_i\})$	$\sum_i w_i [L_i] + \exp(\sum_i w_i \log(\mathbb{L}_i))$	$\sum_i w_i [L_i] + (\sum_i w_i \mathbb{L}_i^{\theta})^{\frac{1}{\theta}}$	$\sum_i w_i [L_i] + \left(\sum_i w_i \mathbb{L}_i^{\frac{\theta}{2}} \right)^{\frac{2}{\theta}}$
$L \oplus K$	$[L] + [K] + \mathbb{L}\mathbb{K}$	$[L] + [K] + (\mathbb{L}^{\theta} + \mathbb{K}^{\theta} - I_n)^{\frac{1}{\theta}}$	$[L] + [K] + \left(\mathbb{L}^{\frac{\theta}{2}} + \mathbb{K}^{\frac{\theta}{2}} - I_n \right)^{\frac{2}{\theta}}$
$t \odot L$	$t[L] + \mathbb{L}^t$	$t[L] + (t\mathbb{L}^{\theta} + (1-t)I_n)^{\frac{1}{\theta}}$	$t[L] + \left(t\mathbb{L}^{\frac{\theta}{2}} + (1-t)I_n \right)^{\frac{2}{\theta}}$

Table 6.9: Riemannian and gyro operators of different metrics on the Cholesky manifold. For the diagonal log metric, $\log(\cdot)$ and $\exp(\cdot)$ are diagonal logarithm and exponentiation.

function, while those under the diagonal log metric are computed using the diagonal logarithm or exponentiation. This indicates that our θ -DPM and $(\theta, \mathbb{M})$ -DBWM may have better numerical stability than the existing diagonal log metric, as logarithm or exponentiation might overly stretch the diagonal elements compared with the power function. The gyro operations under our θ -DPM and $(\theta, \mathbb{M})$ -DBWM also have numerical advantages over those under the diagonal log metric. The former are based on linear operations combined with power and its inverse, causing relatively minor changes to the input magnitude, while the gyro operations under the diagonal log metric are based on products or powers, resulting in more noticeable alterations to the input magnitude.

6.3.4 Geometries on the SPD Manifold

This section discusses the Riemannian metrics on the SPD manifold via the Cholesky decomposition. We first review some basic properties of the Cholesky decomposition, followed by the SPD metrics.

The Cholesky decomposition, denoted by $\text{Chol}(\cdot) : \mathcal{S}_{++}^n \rightarrow \mathcal{L}_{++}^n$, is a diffeomorphism [137]. Therefore, it can pull back the Riemannian and gyro-structures from the Cholesky manifold $\mathcal{L}_{++}^n$ to the SPD manifold $\mathcal{S}_{++}^n$. We call the pullbacks of θ -DPM and $(\theta, \mathbb{M})$ -DBWM through the Cholesky decomposition the Power-Cholesky Metric (θ -PCM) and Bures–Wasserstein–Cholesky Metric $((\theta, \mathbb{M})$ -BWCM), respectively. Then, the Riemannian operators under θ -PCM and $(\theta, \mathbb{M})$ -BWCM can be obtained from the properties of Riemannian isometries (Thm. 33).

Operators. Let $\mathcal{C} \in \{\theta\text{-DPM}, (\theta, \mathbb{M})\text{-DBWM}\}$ and $\mathcal{S} \in \{\theta\text{-PCM}, (\theta, \mathbb{M})\text{-BWCM}\}$. We denote the Riemannian logarithm, exponential map, geodesic, parallel transport along the geodesic, geodesic distance, weighted Fréchet mean, gyroaddition, and scalar

gyromultiplication on $\{\mathcal{S}_{++}^n, g^{\mathcal{S}}\}$ by $\text{Log}^{\mathcal{S}}, \text{Exp}^{\mathcal{S}}, \gamma^{\mathcal{S}}, \text{PT}^{\mathcal{S}}, d^{\mathcal{S}}(\cdot, \cdot), \text{WFM}^{\mathcal{S}}, \oplus^{\mathcal{S}},$ and $\odot^{\mathcal{S}},$ respectively, while $\text{Log}^{\mathcal{C}}, \text{Exp}^{\mathcal{C}}, \gamma^{\mathcal{C}}, \text{PT}^{\mathcal{C}}, d^{\mathcal{C}}(\cdot, \cdot), \text{WFM}^{\mathcal{C}}, \oplus^{\mathcal{C}},$ and $\odot^{\mathcal{C}}$ are their counterparts on $\{\mathcal{L}_{++}^n, g^{\mathcal{C}}\}.$ For $P, Q \in \mathcal{S}_{++}^n, V, W \in T_P \mathcal{S}_{++}^n,$ and $\{P_i \in \mathcal{S}_{++}^n\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$ satisfying $w_i > 0$ for all i and $\sum_{i=1}^N w_i = 1,$ we have the following Riemannian and gyro operators:

$$\gamma_{(P,V)}^{\mathcal{S}}(t) = \text{Chol}^{-1} \left(\gamma_{(L,\tilde{V})}^{\mathcal{C}}(t) \right), \quad (6.60)$$

$$\text{Log}_P^{\mathcal{S}}(Q) = (\text{Chol}_{*,P})^{-1} \left(\text{Log}_L^{\mathcal{C}}(K) \right), \quad (6.61)$$

$$\text{Exp}_P^{\mathcal{S}}(V) = \text{Chol}^{-1} \left(\text{Exp}_L^{\mathcal{C}}(\tilde{V}) \right), \quad (6.62)$$

$$\text{PT}_{P \rightarrow Q}^{\mathcal{S}}(V) = (\text{Chol}_{*,Q})^{-1} \left(\text{PT}_{L \rightarrow K}^{\mathcal{C}}(\tilde{V}) \right), \quad (6.63)$$

$$d^{\mathcal{S}}(P, Q) = d^{\mathcal{C}}(L, K), \quad (6.64)$$

$$\text{WFM}^{\mathcal{S}}(\{w_i\}, \{P_i\}) = \text{Chol}^{-1} \left(\text{WFM}^{\mathcal{C}}(\{w_i\}, \{L_i\}) \right), \quad (6.65)$$

$$P \oplus^{\mathcal{S}} Q = \text{Chol}^{-1}(L \oplus^{\mathcal{C}} K), \quad (6.66)$$

$$t \odot^{\mathcal{S}} P = \text{Chol}^{-1}(t \odot^{\mathcal{C}} L), \quad (6.67)$$

where $P = LL^{\top}, Q = KK^{\top},$ and $P_i = L_i L_i^{\top}$ are Cholesky decompositions. Here, $\tilde{V} = \text{Chol}_{*,P}(V)$ is the Cholesky differential as defined in Sec. 2.8.1. Besides, when $\mathcal{C}$ is the diagonal log metric, the above recovers the Riemannian and gyro-structures under LCM.

The above gyro operations, when well-defined, also satisfy the gyrovector-space axioms.

Theorem 177. $\{\mathcal{S}_{++}^n, \oplus^{\mathcal{S}}, \odot^{\mathcal{S}}\}$ satisfies all the axioms of gyrovector spaces.

Remark 178. As discussed in Sec. 6.3.3.5, the Cholesky metrics θ -DPM and $(\theta, \mathbb{M})$ -DBWM are more numerically stable than the existing diagonal log metric. As pullback metrics through the Cholesky decomposition, our θ -PCM and $(\theta, \mathbb{M})$ -BWCM therefore preserve the advantage of numerical stability over the existing LCM. Besides, all the Riemannian operators have closed-form expressions and are easy to use, as the differential maps of the Cholesky decomposition can be easily calculated.

6.3.5 Applications to SPD Neural Networks

The closed-form operators derived in Sec. 6.3.4 make the proposed metrics directly applicable to SPD neural networks. In this section, we apply the proposed SPD metrics θ -PCM and $(\theta, \mathbb{M})$ -BWCM to build MLR classifiers and residual blocks on the SPD manifold.

MLR. The Euclidean point-to-hyperplane formulation and its Riemannian extension are developed in Chapter 4. Substituting the operators of the proposed metrics into that formulation gives the following SPD MLRs.

Theorem 179. $\Downarrow$ Given an input SPD matrix $S \in \mathcal{S}_{++}^n$, the C -class SPD MLRs under θ -PCM and $(\theta, \mathbb{M})$ -BWCM are

$$\theta\text{-PCM}: p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[\begin{aligned} &\langle [K] - [L_k], [A_k] \rangle \\ &+ \frac{1}{2\theta} \langle \mathbb{K}^\theta - \mathbb{L}_k^\theta, \mathbb{A}_k \rangle \end{aligned} \right], \quad (6.68)$$

$$(\theta, \mathbb{M})\text{-BWCM}: p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[\begin{aligned} &\langle [K] - [L_k], [A_k] \rangle \\ &+ \frac{1}{4\theta} \langle \mathbb{K}^{\frac{\theta}{2}} - \mathbb{L}_k^{\frac{\theta}{2}}, \mathbb{M}^{-1} \mathbb{A}_k \rangle \end{aligned} \right], \quad (6.69)$$

where $S = KK^\top$ and $P_k = L_k L_k^\top$ are Cholesky decompositions. The parameters are $P_k \in \mathcal{S}_{++}^n$ and $A_k \in \text{LT}^n$ for each class $k = 1, \dots, C$.

Residual blocks. The general construction and the specialized SPD residual-block expression are reviewed in Sec. A.2.5. The only component that varies across metrics is the Riemannian exponential map; substituting the operators derived above therefore gives residual blocks under the proposed metrics.

6.3.6 Experiments

We first compare our metrics against the popular AIM, LEM, and LCM when building SPD MLR classifiers and residual blocks. Then, we evaluate the proposed metrics through numerical experiments. More details on data sets and experimental settings are provided in Secs. A.1 and A.3.8.

6.3.6.1 Riemannian Classifiers

Following the SPD learning setup in Sec. 4.4 and Huang and Van Gool [106], we adopt the Radar data set [31] for radar signal classification, and the HDM05 [153] and FPHA [80] data sets for human action recognition. We compare the SPD MLRs under our

6.3. Product Cholesky Metrics

(a) Radar			(b) HDM05						(c) FPHA			
Metric	Acc	Time	1-Block		2-Block		3-Block		Metric	Acc	Time	
			Acc	Time	Acc	Time	Acc	Time				
AIM	94.53 $\pm$ 0.95	0.80	AIM	58.07 $\pm$ 0.64	17.32	60.72 $\pm$ 0.62	18.75	61.14 $\pm$ 0.94	19.23	AIM	85.57 $\pm$ 0.50	7.14
LEM	93.55 $\pm$ 1.21	0.76	LEM	56.97 $\pm$ 0.61	2.21	60.69 $\pm$ 1.02	2.92	60.28 $\pm$ 0.91	3.50	LEM	85.90 $\pm$ 0.47	0.98
LCM	93.49 $\pm$ 1.25	0.72	LCM	60.69 $\pm$ 1.89	1.83	62.61 $\pm$ 1.46	2.40	62.33 $\pm$ 2.15	2.90	LCM	86.37 $\pm$ 0.59	0.74
θ -PCM	95.79 $\pm$ 0.38	0.72	θ -PCM	62.51 $\pm$ 1.65	1.58	63.66 $\pm$ 1.30	2.29	65.75 $\pm$ 2.86	2.76	θ -PCM	89.40 $\pm$ 0.13	0.69
θ -BWCM	93.93 $\pm$ 0.79	0.71	θ -BWCM	62.71 $\pm$ 0.88	1.64	64.52 $\pm$ 0.56	2.27	67.40 $\pm$ 0.90	2.87	θ -BWCM	86.27 $\pm$ 0.60	0.70

Table 6.10: SPD MLRs under different metrics on the SPDNet backbone. The best results are **bold**.

metrics with those under AIM, LEM, and LCM in Thm. 117. Following Sec. 4.4 and Nguyen and Yang [159], we adopt SPDNet [106] and GyroSPD [159] as two backbones, both mimicking feedforward neural networks. For simplicity, we set $\mathbb{M}$ in $(\theta, \mathbb{M})$ -BWCM to the identity matrix.

SPDNet. On HDM05, we further evaluate architectures with up to three transformation blocks. Tab. 6.10 reports the five-fold accuracy and training time per epoch, from which we draw the following observations.

- **Effectiveness.** Our metrics generally yield higher accuracy than their counterparts. Notably, they outperform LCM, although both originate from the Cholesky product structure. This improvement is attributed to the fact that the diagonal logarithm and exponentiation in LCM tend to overly stretch the diagonal entries, *i.e.*, eigenvalues of the Cholesky factors, whereas our diagonal power transformation achieves a more balanced scaling.
- **Efficiency.** Our metrics substantially reduce computational cost compared to AIM, remain faster than LEM, and achieve efficiency comparable to LCM. Together with their superior accuracy, these results highlight the dual advantages of our approach in both effectiveness and efficiency.

GyroSPD. Tab. 6.11 reports the results on the GyroSPD backbone, which consists of a single gyrotranslation layer followed by an SPD MLR. Similar to the SPDNet results, our metrics achieve comparable or superior performance to LCM across

Metric	Radar		HDM05		FPHA	
	Acc	Time	Acc	Time	Acc	Time
AIM	96.80 $\pm$ 0.59	1.23	66.05 $\pm$ 1.80	21.65	85.77 $\pm$ 0.52	11.48
LEM	96.58 $\pm$ 0.27	1.18	66.42 $\pm$ 0.47	2.02	85.87 $\pm$ 0.79	1.22
LCM	96.29 $\pm$ 0.53	1.12	68.37 $\pm$ 0.66	1.66	89.83 $\pm$ 0.28	0.98
θ -PCM	97.04 $\pm$ 0.64	1.18	71.93 $\pm$ 1.21	1.51	91.17 $\pm$ 0.30	1.00
θ -BWCM	96.21 $\pm$ 0.25	1.05	72.74 $\pm$ 0.43	1.58	91.00 $\pm$ 0.11	0.96

Table 6.11: SPD MLRs on the GyroSPD backbone.

all data sets while maintaining comparable efficiency. On HDM05 and FPHA, both proposed metrics deliver higher accuracy with lower runtime than AIM and LEM. On Radar, PCM achieves the highest accuracy, while BWCM has the lowest runtime.

6.3.6.2 Riemannian Residual Blocks

Riemannian ResNet (RResNet) was introduced by Katsman et al. [118]. Its backbone architecture largely follows SPDNet. The key difference lies in the head: while SPDNet directly applies a classification layer, RResNet inserts a residual block before the classification head. Accordingly, we adopt the following classification head

under each metric: $\text{Log}_{I_n} + \text{FC} + \text{softmax}$. Since the Riemannian exponential is similar for θ -PCM and $(\theta, \mathbb{M})$ -BWCM, we focus on θ -PCM and compare it against AIM, LEM, and LCM in constructing RResNet. Tab. 6.12 reports the best results across three trials, showing that our metric consistently achieves superior accuracy while maintaining comparable efficiency.

Metric	Radar		HDM05		FPFA	
	Acc	Time	Acc	Time	Acc	Time
AIM	96.4	1.02	57.01	1.14	87.33	0.72
LEM	97.07	0.81	67.52	0.52	86.17	0.32
LCM	97.07	0.85	66.27	0.63	86.83	0.49
θ -PCM	97.87	0.85	68.05	0.63	88.33	0.48

Table 6.12: Results on residual blocks.

6.3.6.3 Numerical Stability

As discussed by Lin [137, p. 16], LCM is more stable than AIM and LEM owing to the numerical advantage of Cholesky decomposition over SVD. Moreover, as highlighted in Thm. 178, the essential distinction between our metrics and LCM lies in the diagonal operations: ours rely on diagonal power, while LCM employs diagonal exponentiation and logarithm. This structural difference grants our metrics stronger numerical stability and robustness compared with LCM, as well as AIM and LEM. To validate this, we evaluate geodesics on the Cholesky manifold. We generate 100,000 synthetic $n \times n$ Cholesky matrices L and tangent vectors $X \in \text{LT}^n$, where each entry is uniformly sampled from $[0, 1]$, and we set the smallest eigenvalue (diagonal entry) of L to ϵ . We test two representative sizes: 3×3 matrices, commonly used in diffusion tensor imaging [10], and 256×256 matrices, typical in computer vision [133, 206]. The deformation parameter θ is set to 1.5, 0.5, and 0.15. For $(\theta, \mathbb{M})$ -DBWM, we set $\mathbb{M} = I_n$. As shown in Tab. 6.13, our θ -DPM and θ -DBWM remain highly stable across a wide range of ϵ . For 3×3 matrices, the diagonal log metric already deteriorates at $\epsilon = 1e^{-3}$, with failure rates increasing rapidly as ϵ decreases. For 256×256 matrices, instability emerges even earlier at $\epsilon = 1e^{-1}$. In contrast, our metrics remain stable down to $\epsilon = 1e^{-30}$ in most cases. The only exception occurs when $\theta = 0.15$, where failures appear around $\epsilon = 1e^{-20}$. This behavior is expected since both θ -DPM and θ -DBWM converge to the diagonal log metric as $\theta \rightarrow 0$, thereby inheriting its instability in this limit. Overall,

6.3. Product Cholesky Metrics

ϵ	3 × 3 for small matrices							256 × 256 for large matrices						
	DLM	$\theta = 1.5$		$\theta = 0.5$		$\theta = 0.15$		DLM	$\theta = 1.5$		$\theta = 0.5$		$\theta = 0.15$	
		DPM	DBWM	DPM	DBWM	DPM	DBWM		DPM	DBWM	DPM	DBWM	DPM	DBWM
$1e^{-1}$	0.62	0	0	0	0	0	0	14.29	0	0	0	0	0	0
$1e^{-2}$	5.70	0	0	0	0	0	0	18.48	0	0	0	0	0	0
$1e^{-3}$	51.32	0	0	0	0	0	0	58.35	0	0	0	0	0	0
$1e^{-4}$	94.34	0	0	0	0	0	0	95.02	0	0	0	0	0	0
$1e^{-5}$	99.39	0	0	0	0	0	0	99.47	0	0	0	0	0	0
$1e^{-10}$	100	0	0	0	0	0	0	100	0	0	0	0	0	0
$1e^{-15}$	100	0	0	0	0	0	0	100	0	0	0	0	0	0
$1e^{-20}$	100	0	0	0	0	0	0.002	100	0	0	0	0	0	0.02
$1e^{-21}$	100	0	0	0	0	0	0.03	100	0	0	0	0	0	0.01
$1e^{-22}$	100	0	0	0	0	0	0.25	100	0	0	0	0	0	0.23
$1e^{-23}$	100	0	0	0	0	0	2.26	100	0	0	0	0	0	2.42
$1e^{-24}$	100	0	0	0	0	0	22.98	100	0	0	0	0	0	23.13
$1e^{-25}$	100	0	0	0	0	0	86.34	100	0	0	0	0	0	86.58
$1e^{-30}$	100	0	0	0	0	0	100	100	0	0	0	0	0	100

Table 6.13: Failure probabilities (%) of geodesics under different metrics with small eigenvalues in $L \in \mathcal{L}_{++}^n$. An output matrix containing any INF or NAN is considered a failure. Here, DLM denotes the diagonal log metric, while DPM and DBWM denote θ -DPM and θ -DBWM, respectively.

these results demonstrate the superior numerical robustness of our metrics.

6.3.6.4 Tensor Interpolation

As shown by Arsigny et al. [9, 10], geodesic interpolation of SPD matrices is important in diffusion tensor imaging. This experiment illustrates geodesic interpolation under different SPD metrics. Let $P = LL^\top$ and $Q = KK^\top$ be the Cholesky decompositions of $P, Q \in \mathcal{S}_{++}^n$. The geodesics connecting P and Q under θ -PCM and $(\theta, \mathbb{M})$ -BWCM are

$$\theta\text{-PCM: } \text{Chol}^{-1} \left[\lfloor L \rfloor + t(\lfloor K \rfloor - \lfloor L \rfloor) + (\mathbb{L}^\theta + t(\mathbb{K}^\theta - \mathbb{L}^\theta))^{\frac{1}{\theta}} \right], \quad (6.70)$$

$$(\theta, \mathbb{M})\text{-BWCM: } \text{Chol}^{-1} \left[\lfloor L \rfloor + t(\lfloor K \rfloor - \lfloor L \rfloor) + \left(\mathbb{L}^{\frac{\theta}{2}} + t(\mathbb{K}^{\frac{\theta}{2}} - \mathbb{L}^{\frac{\theta}{2}}) \right)^{\frac{2}{\theta}} \right]. \quad (6.71)$$

As the geodesics under θ -PCM and $(\theta, \mathbb{M})$ -BWCM have similar expressions, we focus on θ -PCM. Fig. 6.3 visualizes the geodesic interpolations on $\mathcal{S}_{++}^3$ under different metrics, including θ -EM, LEM, AIM, BWCM, LCM, and θ -PCM. Tab. 6.14 presents the associated determinants of the interpolated SPD matrices. We can make the following observations.

- (1) The standard Euclidean metric (1-EM) exhibits a significant swelling effect, where the maximal determinant of interpolation is much larger than the determinants of the starting and end points. Although matrix power can mitigate the swelling effect, θ -EM still suffers from swelling.
- (2) We find that BWCM also demonstrates a clear swelling effect. In contrast, LCM,

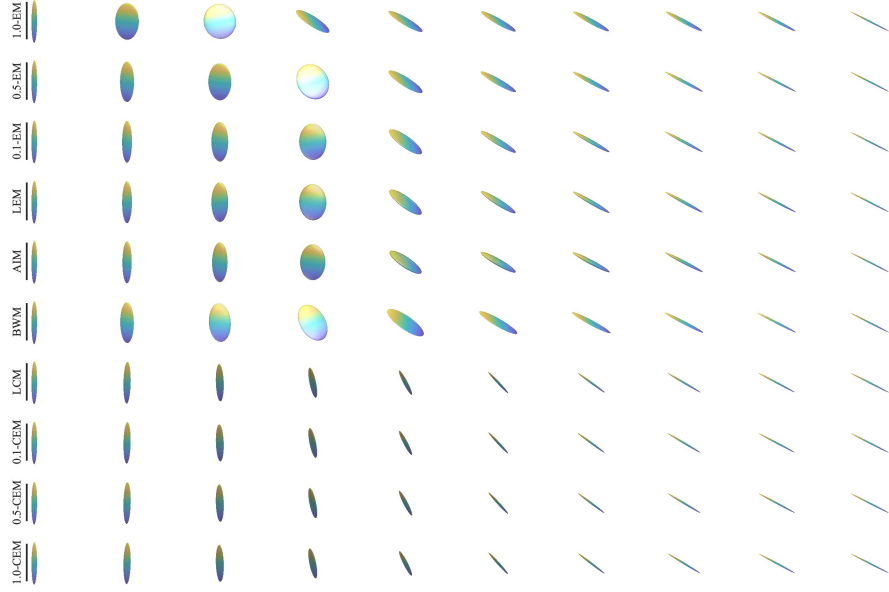


Figure 6.3: Geodesic interpolation of SPD matrices under different Riemannian metrics. Each 3×3 SPD matrix can be visualized as an ellipsoid [9]. The two endpoints are fixed across all metrics.

AIM, and LEM show no swelling effect.

- (3) The trivial PCM ($\theta = 1$) considerably mitigates the swelling effect compared to the Euclidean metric, but it still exhibits some level of swelling. However, by introducing Cholesky power deformation, θ -PCM effectively reduces the swelling effect. Notably, the swelling effect of θ -PCM is significantly weaker than that of θ -EM under the same θ .
- (4) Our PCM shows an interpolation visually similar to that under LCM. This suggests that our metric retains some practical potential of LCM but with better numerical stability (as demonstrated in Sec. 6.3.6.3).

6.3.6.5 Asymptotic Complexity

We investigate the scalability of our PCM and BWCM in comparison with five existing SPD metrics, namely AIM, LEM, LCM, PEM, and BWM. We use SPD MLR as a representative application. We first analyze the asymptotic complexity of each SPD MLR. We then complement this analysis with synthetic experiments that measure the actual runtime of a single SPD MLR training step across different matrix dimensions.

6.3. Product Cholesky Metrics

Metric	The determinant of the i -th interpolation									
	0	1	2	3	4	5	6	7	8	9
1.0-EM	3.07	104.86	182.09	234.38	261.35	262.64	237.86	186.64	108.61	3.38
0.5-EM	3.07	18.67	39.93	59.53	71.96	73.79	64.14	45.01	21.73	3.38
0.1-EM	3.07	4.25	5.42	6.38	6.96	7.05	6.62	5.75	4.6	3.38
LEM	3.07	3.1	3.14	3.17	3.2	3.24	3.27	3.31	3.34	3.38
AIM	3.07	3.1	3.14	3.17	3.2	3.24	3.27	3.31	3.34	3.38
BWM	3.07	15.32	32.04	48.14	59.22	62.09	55.33	39.93	19.98	3.38
LCM	3.07	3.1	3.14	3.17	3.2	3.24	3.27	3.31	3.34	3.38
0.1-PCM	3.07	3.15	3.23	3.29	3.34	3.37	3.39	3.4	3.4	3.38
0.5-PCM	3.07	3.35	3.59	3.79	3.91	3.97	3.94	3.83	3.64	3.38
1.0-PCM	3.07	3.6	4.07	4.46	4.72	4.83	4.76	4.49	4.03	3.38

Table 6.14: Swelling effects of geodesic SPD interpolations. Deeper greens indicate greater swelling.

Metric	Num. spectral matrix functions	Num. Cholesky decompositions
AIM	$1 + 2C$	0
LEM	$1 + C$	0
LCM	0	$1 + C$
PEM	$1 + C$	0
BWM	$1 + 3C$	C
PCM	0	$1 + C$
BWCM	0	$1 + C$

Table 6.15: Number of matrix functions required per sample for a C -class SPD MLR. Spectral matrix functions include matrix logarithm, matrix power, and the Lyapunov operator.

As shown in Thms. 117 and 179, the C -class SPD MLRs under different metrics for the input $S \in \mathcal{S}_{++}^n$ are

$$\begin{aligned}
\text{AIM: } p(y = k \mid S) &\propto \exp \left[\left\langle \log \left(P_k^{-\frac{1}{2}} S P_k^{-\frac{1}{2}} \right), A_k \right\rangle \right], \\
\text{LEM: } p(y = k \mid S) &\propto \exp [\langle \log(S) - \log(P_k), A_k \rangle], \\
\text{LCM: } p(y = k \mid S) &\propto \exp \left[\left\langle \begin{matrix} [K] - [L_k] + \log(\mathbb{K}) \\ -\log(\mathbb{L}_k) \end{matrix}, [A_k] + \frac{1}{2} \mathbb{A}_k \right\rangle \right], \\
\text{PEM: } p(y = k \mid S) &\propto \exp \left[\frac{1}{\theta} \langle S^\theta - P_k^\theta, A_k \rangle \right], \\
\text{BWM: } p(y = k \mid S) &\propto \exp \left[\frac{1}{2} \left\langle (P_k S)^{\frac{1}{2}} + (S P_k)^{\frac{1}{2}} - 2P_k, \mathcal{L}_{P_k}(L_k A_k L_k^\top) \right\rangle \right], \\
\theta\text{-PCM: } p(y = k \mid S) &\propto \exp \left[\langle [K] - [L_k], [A_k] \rangle + \frac{1}{2\theta} \langle \mathbb{K}^\theta - \mathbb{L}_k^\theta, \mathbb{A}_k \rangle \right],
\end{aligned}$$

$$(\theta, \mathbb{M})\text{-BWCM} : p(y = k \mid S) \propto \exp \left[\begin{aligned} &\langle \lfloor K \rfloor - \lfloor L_k \rfloor, \lfloor A_k \rfloor \rangle \\ &+ \frac{1}{4\theta} \left\langle \mathbb{K}^{\frac{\theta}{2}} - \mathbb{L}_k^{\frac{\theta}{2}}, \mathbb{M}^{-1} \mathbb{A}_k \right\rangle \end{aligned} \right],$$

where $P_k \in \mathcal{S}_{++}^n$ and $A_k \in \mathcal{S}^n$ are MLR weights, $\log(\cdot)$ is the matrix logarithm, and $\mathcal{L}_P[V]$ is the solution to the matrix linear system $\mathcal{L}_P[V]P + P\mathcal{L}_P[V] = V$, known as the Lyapunov operator.

Analysis. Tab. 6.15 summarizes the number of spectral and Cholesky matrix functions required by each SPD MLR. Cholesky decomposition requires $O(1/3n^3)$ flops, while eigendecomposition costs $O(9n^3)$ flops [83, Algs. 4.2.3 and 8.3.3]. Combining these counts, Tab. 6.16 reports the resulting asymptotic per-sample complexity for each metric. Cholesky-based metrics (LCM, PCM, BWCM) are asymptotically more efficient than the eigen-based metrics (LEM, PEM, AIM, and BWM), with AIM and BWM being the slowest among the considered methods. In addition, PCM and BWCM can be practically more efficient than LCM, since diagonal powers are cheaper to compute than diagonal logarithms.

Setup. To validate the asymptotic complexity in Tab. 6.16, we measure the average wall-clock time of a single forward-backward training step of an SPD MLR classifier as the matrix dimension increases. The model consists of a single SPD MLR layer with 50 output classes followed by a cross-entropy loss. For each dimension $n \in \{32, 64, 128, 256, 512\}$, we randomly generate a batch of 30 $n \times n$ SPD matrices. In each run, we perform one forward and one backward pass and record the total runtime of this step. For PEM, we set the matrix power to 0.5.

Metric	Asymptotic complexity
AIM	$O(9(1 + 2C)n^3)$
LEM	$O(9(1 + C)n^3)$
LCM	$O(\frac{1+C}{3}n^3)$
PEM	$O(9(1 + C)n^3)$
BWM	$O((9(1 + 3C) + \frac{C}{3})n^3)$
PCM	$O(\frac{1+C}{3}n^3)$
BWCM	$O(\frac{1+C}{3}n^3)$

Table 6.16: Asymptotic per-sample complexity of a C -class SPD MLR for an $n \times n$ input SPD matrix.

Results. As reported in Tab. 6.17, our PCM and BWCM are the fastest metrics across all tested dimensions, and the gap becomes particularly pronounced in the high-dimensional case. For small and medium scales (32 and 64), LCM, PCM, and BWCM have very similar runtimes and all are clearly faster than AIM, LEM, PEM, and BWM. When the dimension increases to 512, AIM and BWM require about 60 and 70 seconds per training step, whereas PCM and BWCM remain within roughly 1.7 seconds. In this setting, PCM and BWCM are even faster than LCM.

Dim	AIM	LEM	LCM	PEM	BWM	PCM	BWCM
32	0.2380	0.0077	0.0046	0.0076	0.2377	0.0040	0.0040
64	1.0139	0.0395	0.0303	0.0473	1.1205	0.0251	0.0225
128	3.6256	0.1832	0.1490	0.1844	4.0674	0.1013	0.1019
256	14.5142	0.7793	0.5833	0.7853	16.5918	0.3848	0.4077
512	60.1918	3.2948	2.5030	3.4357	70.8647	1.7553	1.7526

Table 6.17: Average runtime (in seconds) of one SPD MLR training step across different matrix dimensions.

6.4 Conclusion

This chapter developed SPD metric design along two complementary routes. The first route proposed a pullback Euclidean framework and used a learnable general matrix logarithm to construct ALEM. Through this pullback construction, ALEM inherits compatible Hilbert-space, abelian Lie-group, and Riemannian structures, together with closed-form Riemannian operators and weighted Fréchet means. We further derived the differentials, gradients, and parameter-update schemes required to learn the logarithm bases. Experiments with applications to SPDNet, LieBN, Riemannian residual blocks, and gyro MLR demonstrate the effectiveness of our metrics.

The second route revealed the product structure of the Cholesky manifold. This structure yielded diagonal power and diagonal Bures–Wasserstein geometries, whose deformed versions converge to the diagonal log metric as the power parameter approaches zero. Transferring these geometries through the Cholesky decomposition produced PCM and BWCM on the SPD manifold. The resulting metrics admit closed-form Riemannian and gyro operators, which directly yield SPD MLR classifiers and residual blocks. Experiments on SPD classifiers and residual networks, together with numerical experiments, supported their effectiveness, efficiency, and numerical robustness.

Together, these routes show that the underlying Riemannian geometry can itself be designed to provide the flexibility, tractability, efficiency, and numerical stability required by Riemannian deep learning.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis studied Riemannian deep learning from three connected perspectives: unified Riemannian module design across manifolds, manifold-specific Riemannian network design, and the design of the underlying Riemannian geometries. The main contributions are summarized as follows:

- Chapters 3 and 4 developed unified *network modules* from geometric structures shared across different manifolds. The normalization chapter first developed LieBN on Lie groups, then introduced pseudo-reductive gyrogroups, a new algebraic structure that generalizes classical gyrogroups and Lie groups, and finally developed GyroBN on this foundation, with LieBN recovered as a special case. The classification development progressed from SPD MLRs obtained through exact evaluation of the point-to-hyperplane infimum under flat pullback Euclidean metrics to a general RMLR based on a Riemannian-trigonometric formulation requiring only a well-defined Riemannian logarithm.
- Chapter 5 developed *manifold-specific Riemannian network designs* by exploiting additional structures. PVNN developed the geometry and core neural layers of the stable PV model, *Hyperbolic Busemann Neural Networks (HBNN)* derived intrinsic and efficient BMLR and BFC layers from Busemann functions and horospheres on the Poincaré and Lorentz models, and CorNet developed correlation MLR, FC, and convolutional layers under five geometries together with accurate Riemannian backpropagation under OLM and LSM.
- Chapter 6 designed the underlying *Riemannian geometries*. ALEM learned pull-

back Euclidean metrics through general matrix logarithms, whereas PCM and BWCM used the product structure of the Cholesky manifold to obtain efficient and numerically stable SPD geometries.

Taken together, these contributions address Riemannian deep learning at the levels of unified network modules, manifold-specific network designs, and underlying Riemannian geometries while balancing intrinsic structure, generality, computational tractability, and numerical stability.

7.2 Future Work

Two directions are particularly promising: representation learning with novel geometries and geometry-aware generative modeling.

- **Novel geometries for complex relational structure.** Hyperbolic spaces provide an effective inductive bias for hierarchical data [164]. Although mixed-curvature products [87, 182] and matrix manifolds [59] have demonstrated the benefit of matching geometry to heterogeneous structure, real systems may also contain more complex relation types and structures. Future work could develop novel geometric structures to encode such complex relationships while retaining efficient and tractable Riemannian computations.
- **Geometry-aware generative modeling.** Geometry arises both in latent representations and in spaces of probability distributions. At the latent level, Riemannian variational autoencoders [114], continuous normalizing flows [148], score-based models [64], and flow matching [43] have shown how manifold geometry can shape generative dynamics, while other work has analyzed the geometry of learned latent spaces [11, 168]. At the distribution level, a growing body of work has explored the geometry of probability distributions through information geometry [56, 63, 57] and optimal transport [96, 58]. A central problem is therefore to balance the geometry of the latent space with the geometry of the distributions evolving on it.

Bibliography

- [1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.
- [2] Bijan Afsari. Riemannian Lp center of mass: Existence, uniqueness, and convexity. In *Proceedings of the American Mathematical Society*, 2011.
- [3] Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016. URL <https://doi.org/10.1007/978-4-431-55978-8>.
- [4] Roy M Anderson and Robert M May. *Infectious Diseases of Humans: Dynamics and Control*. Oxford University Press, 1991.
- [5] Tsuyoshi Ando, Chi-Kwong Li, and Roy Mathias. Geometric means. *Linear algebra and its applications*, 385:305–334, 2004. URL <https://doi.org/10.1016/j.laa.2003.11.019>.
- [6] Ilya Archakov and Peter Reinhard Hansen. A new parametrization of correlation matrices. *Econometrica*, 89(4):1699–1715, 2021.
- [7] Ilya Archakov and Peter Reinhard Hansen. A canonical representation of block matrices with applications to covariance and correlation matrices. *Review of Economics and Statistics*, 106(4):1099–1113, 2024.
- [8] Marc Arnaudon, Frédéric Barbaresco, and Le Yang. Riemannian medians and means with applications to radar signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 7(4):595–604, 2013. URL <https://doi.org/10.1109/JSTSP.2013.2261798>.
- [9] Vincent Arsigny, Pierre Fillard, Xavier Pennec, and Nicholas Ayache. *Fast and simple computations on tensors with log-Euclidean metrics*. PhD thesis, INRIA, 2005. URL https://doi.org/10.1007/11566465_15.

- [10] Vincent Arsigny, Pierre Fillard, Xavier Pennec, and Nicholas Ayache. Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM journal on matrix analysis and applications*, 29(1):328–347, 2007.
- [11] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. *ICLR*, 2018.
- [12] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- [13] Gregor Bachmann, Gary Bécigneul, and Octavian Ganea. Constant curvature graph convolutional networks. In *ICML*, 2020.
- [14] Frédéric Barbaresco. Gaussian distributions on the space of symmetric positive definite matrices from Souriau’s Gibbs state for Siegel domains by coadjoint orbit and moment map. In *Geometric Science of Information: 5th International Conference*, 2021.
- [15] Ahmad Bdeir, Kristian Schwethelm, and Niels Landwehr. Fully hyperbolic convolutional neural networks for computer vision. In *ICLR*, 2024.
- [16] Ahmad Bdeir, Johannes Burchert, Lars Schmidt-Thieme, and Niels Landwehr. Robust hyperbolic learning with curvature-aware optimization. In *NeurIPS*, 2025.
- [17] Gary Becigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *ICLR*, 2019.
- [18] Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *ICLR*, 2019.
- [19] Thomas Bendokat, Ralf Zimmermann, and P-A Absil. A Grassmann manifold handbook: Basic geometry and computational aspects. *Advances in Computational Mathematics*, 50(1):1–51, 2024.
- [20] Rajendra Bhatia. *Positive Definite Matrices*. Princeton University Press, 2007.
- [21] Rajendra Bhatia. *Matrix analysis*, volume 169 of *Graduate Texts in Mathematics*. Springer New York, 2013. doi: 10.1007/978-1-4612-0653-8.
- [22] Rajendra Bhatia, Tanvi Jain, and Yongdo Lim. On the Bures-Wasserstein distance between positive definite matrices. *Expositiones Mathematicae*, 37(2):165–191, 2019.

-
- [23] Victoria Bloom, Dimitrios Makris, and Vasileios Argyriou. G3D: A gaming action dataset and real time action recognition evaluation framework. In *CVPR Workshops*, 2012.
 - [24] Clément Bonet, Lucas Drumetz, and Nicolas Courty. Sliced-Wasserstein distances and flows on Cartan-Hadamard manifolds. *JMLR*, 2025.
 - [25] Silvere Bonnabel. Stochastic gradient descent on Riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013. URL <https://arxiv.org/abs/1111.5280>.
 - [26] Silvere Bonnabel, Anne Collard, and Rodolphe Sepulchre. Rank-preserving geometric means of positive semi-definite matrices. *Linear Algebra and its Applications*, 438(8):3202–3216, 2013.
 - [27] Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.
 - [28] Nicolas Boumal and P-A Absil. A discrete regression method on manifolds and its application to data on $so(n)$. *IFAC Proceedings Volumes*, 44(1):2284–2289, 2011.
 - [29] Martin R Bridson and André Haefliger. *Metric spaces of non-positive curvature*, volume 319. Springer Science & Business Media, 1999.
 - [30] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.
 - [31] Daniel Brooks, Olivier Schwander, Frédéric Barbaresco, Jean-Yves Schneider, and Matthieu Cord. Riemannian batch normalization for SPD neural networks. In *NeurIPS*, 2019.
 - [32] Daniel A Brooks, Olivier Schwander, Frédéric Barbaresco, Jean-Yves Schneider, and Matthieu Cord. Exploring complex time-series representations for Riemannian machine learning of radar data. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3672–3676. IEEE, 2019. URL <https://doi.org/10.1109/ICASSP.2019.8683056>.

- [33] James W Cannon, William J Floyd, Richard Kenyon, and Walter R Parry. Hyperbolic geometry. In Silvio Levy, editor, *Flavors of Geometry*, volume 31, pages 59–115. Cambridge University Press, 1997.
- [34] Rudrasis Chakraborty. ManifoldNorm: Extending normalizations on Riemannian manifolds. *arXiv preprint arXiv:2003.13869*, 2020.
- [35] Rudrasis Chakraborty and Baba C Vemuri. Statistics on the Stiefel manifold: theory and applications. *The Annals of Statistics*, 47(1):415–438, 2019.
- [36] Rudrasis Chakraborty, Chun-Hao Yang, Xingjian Zhen, Monami Banerjee, Derek Archer, David Vaillancourt, Vikas Singh, and Baba Vemuri. A statistical recurrent model on the manifold of symmetric positive definite matrices. In *NeurIPS*, 2018.
- [37] Rudrasis Chakraborty, Jose Bouza, Jonathan H Manton, and Baba C Vemuri. Manifoldnet: A deep neural network for manifold-valued data with applications. *IEEE TPAMI*, 2020.
- [38] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. *NeurIPS*, 2019.
- [39] Ines Chami, Albert Gu, Dat P Nguyen, and Christopher Ré. HoroPCA: Hyperbolic dimensionality reduction via horospherical projections. In *ICML*, 2021.
- [40] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *CVPR*, 2019.
- [41] Kai-Xuan Chen, Jie-Yi Ren, Xiao-Jun Wu, and Josef Kittler. Covariance descriptors on a Gaussian manifold and their application to image set classification. *Pattern Recognition*, 107:107463, 2020. doi: 10.1016/j.patcog.2020.107463.
- [42] Kaixuan Chen, Jie Song, Shunyu Liu, Na Yu, Zunlei Feng, Gengshi Han, and Mingli Song. Distribution knowledge embedding for graph pooling. *IEEE TKDE*, 2023.
- [43] Ricky TQ Chen and Yaron Lipman. Flow matching on general geometries. In *ICLR*, 2024.
- [44] Tianyu Chen, Xingcheng Fu, Yisen Gao, Haodong Qian, Yuecen Wei, Kun Yan, Haoyi Zhou, and Jianxin Li. Galaxy walker: Geometry-aware VLMs for galaxy-scale understanding. In *CVPR*, 2025.

-
- [45] Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. Fully hyperbolic neural networks. In *ACL*, 2022.
 - [46] Yuxin Chen, Ziqi Zhang, Chunfeng Yuan, Bing Li, Ying Deng, and Weiming Hu. Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *ICCV*, 2021.
 - [47] Ziheng Chen, Tianyang Xu, Xiao-Jun Wu, Rui Wang, Zhiwu Huang, and Josef Kittler. Riemannian local mechanism for SPD neural networks. In *AAAI*, 2023.
 - [48] Ziheng Chen, Tianyang Xu, Xiao-Jun Wu, Rui Wang, and Josef Kittler. Hybrid Riemannian graph-embedding metric learning for image set classification. *IEEE Transactions on Big Data*, 9(1):75–92, 2023. doi: 10.1109/TBDATA.2021.3113084.
 - [49] Ziheng Chen, Yue Song, Gaowen Liu, Ramana Rao Kompella, Xiaojun Wu, and Nicu Sebe. Riemannian multinomial logistics regression for SPD neural networks. In *CVPR*, 2024.
 - [50] Ziheng Chen, Yue Song, Yunmei Liu, and Nicu Sebe. A Lie group approach to Riemannian batch normalization. In *ICLR*, 2024.
 - [51] Ziheng Chen, Yue Song, Rui Wang, Xiao-Jun Wu, and Nicu Sebe. RMLR: Extending multinomial logistic regression into general geometries. In *NeurIPS*, 2024.
 - [52] Ziheng Chen, Yue Song, Xiao-Jun Wu, and Nicu Sebe. Gyrogroup batch normalization. In *ICLR*, 2025.
 - [53] Ziheng Chen, Yue Song, Xiaojun Wu, Gaowen Liu, and Nicu Sebe. Understanding matrix function normalizations in covariance pooling through the lens of Riemannian geometry. In *ICLR*, 2025.
 - [54] Ziheng Chen, Xiao-Jun Wu, Bernhard Schölkopf, and Nicu Sebe. Riemannian batch normalization: A gyro approach. *arXiv preprint arXiv:2509.07115*, 2025.
 - [55] Ziheng Chen, Xiaojun Wu, Bernhard Schölkopf, and Nicu Sebe. Building transformation layers for Riemannian neural networks, 2025. URL <https://openreview.net/forum?id=1tJVBCpVD0>.
 - [56] Chaoran Cheng, Jiahan Li, Jian Peng, and Ge Liu. Categorical flow matching on statistical manifolds. In *NeurIPS*, 2024.

- [57] Chaoran Cheng, Jiahao Li, Jiajun Fan, and Ge Liu. α -flow: A unified framework for continuous-state discrete flow matching models, 2025. URL <https://arxiv.org/abs/2504.10283>.
- [58] Jaemoo Choi, Jaewoong Choi, and Myungjoo Kang. Scalable Wasserstein gradient flow for generative modeling through unbalanced optimal transport. In *ICML*, 2024.
- [59] Calin Cruceru, Gary Bécigneul, and Octavian-Eugen Ganea. Computationally tractable Riemannian manifolds for graph embeddings. In *AAAI*, 2021.
- [60] Jindou Dai, Yuwei Wu, Zhi Gao, and Yunde Jia. A hyperbolic-to-hyperbolic graph convolutional network. In *CVPR*, 2021.
- [61] Tingting Dan, Ziquan Wei, Won Hwa Kim, and Guorong Wu. Exploring the enigma of neural dynamics through a scattering-transform mixer landscape for Riemannian manifold. In *ICML*, 2024.
- [62] Paul David and Weiqing Gu. A Riemannian structure for correlation matrices. *Operators and Matrices*, 13(3):607–627, 2019.
- [63] Oscar Davis, Samuel Kessler, Mircea Petrache, Ismail Ilkan Ceylan, Michael Bronstein, and Avishek Joey Bose. Fisher flow matching for generative modeling over discrete data. In *NeurIPS*, 2024.
- [64] Valentin De Bortoli, Emile Mathieu, Michael Hutchinson, James Thornton, Yee Whye Teh, and Arnaud Doucet. Riemannian score-based generative modelling. In *NeurIPS*, 2022.
- [65] Thibault de Surrél, Sylvain Chevallier, Fabien Lotte, and Florian Yger. Geometry-aware visualization of high dimensional symmetric positive definite matrices. *TMLR*, 2025.
- [66] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [67] Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. Hyperbolic image-text representations. In *ICML*, 2023.

-
- [68] Abhinav Dhall, Amanjot Kaur, Roland Goecke, and Tom Gedeon. Emotiw 2018: Audio-video, student engagement and group-level affect prediction. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, pages 653–656, 2018. URL <https://doi.org/10.1145/3242969.3264993>.
- [69] Manfredo Perdigao Do Carmo and J Flaherty Francis. *Riemannian Geometry*, volume 6. Springer, 1992.
- [70] Ian L Dryden, Xavier Pennec, and Jean-Marc Peyrat. Power Euclidean metrics for covariance matrices with application to diffusion tensor imaging. *arXiv preprint arXiv:1009.3045*, 2010. URL <https://arxiv.org/abs/1009.3045>.
- [71] Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.
- [72] Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. Hyperbolic vision transformers: Combining improvements in metric learning. In *CVPR*, 2022.
- [73] Xiran Fan, Chun-Hao Yang, and Baba Vemuri. Horospherical decision boundaries for large margin classification in hyperbolic space. In *NeurIPS*, 2023.
- [74] Maurice Fréchet. Les éléments aléatoires de nature quelconque dans un espace distancié. *Annales de l’institut Henri Poincaré*, 10(4):215–310, 1948.
- [75] Xingcheng Fu, Yisen Gao, Yuecen Wei, Qingyun Sun, Hao Peng, Jianxin Li, and Xianxian Li. Hyperbolic geometric latent diffusion model for graph generation. In *ICML*, 2024.
- [76] Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *NeurIPS*, 2018.
- [77] Zhi Gao, Yuwei Wu, Mehrtash Harandi, and Yunde Jia. A robust distance measure for similarity-based classification on the spd manifold. *IEEE TNNLS*, 2019.
- [78] Zhi Gao, Yuwei Wu, Yunde Jia, and Mehrtash Harandi. Curvature generation in curved spaces for few-shot learning. In *ICCV*, 2021.
- [79] Zhi Gao, Chen Xu, Feng Li, Yunde Jia, Mehrtash Harandi, and Yuwei Wu. Exploring data geometry for continual learning. In *CVPR*, 2023.

- [80] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with RGB-D videos and 3D hand pose annotations. In *CVPR*, 2018.
- [81] Mina Ghadimi Atigh, Martin Keller-Ressel, and Pascal Mettes. Hyperbolic busermann learning with ideal prototypes. In *NeurIPS*, 2021.
- [82] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS*, 2011.
- [83] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. JHU press, 2013.
- [84] Alexandre Gramfort. MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience*, 7, 2013.
- [85] Karish Grover, Geoffrey J Gordon, and Christos Faloutsos. CurvGAD: Leveraging curvature for enhanced graph anomaly detection. In *ICML*, 2025.
- [86] Karish Grover, Haiyang Yu, Xiang Song, Qi Zhu, Han Xie, Vassilis N Ioannidis, and Christos Faloutsos. Spectro-Riemannian graph neural networks. In *ICLR*, 2025.
- [87] Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. Learning mixed-curvature representations in product spaces. In *ICLR*, 2019.
- [88] Nicolas Guigui, Nina Miolane, and Xavier Pennec. Introduction to Riemannian geometry and geometric statistics: From basic theory to implementation with Geomstats. *Foundations and Trends in Machine Learning*, 16(3):329–493, 2023. doi: 10.1561/22000000098.
- [89] Johann Guilleminot and Christian Soize. Generalized stochastic approach for constitutive equation in linear elasticity: a random matrix model. *International Journal for Numerical Methods in Engineering*, 90(5):613–635, 2012. URL <https://doi.org/10.1002/nme.3338>.
- [90] Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. Hyperbolic attention networks. In *ICLR*, 2019.
- [91] Yunhui Guo, Xudong Wang, Yubei Chen, and Stella X Yu. Clipped hyperbolic classifiers are super-hyperbolic classifiers. In *CVPR*, 2022.

-
- [92] Andi Han, Bamdev Mishra, Pratik Kumar Jawanpuria, and Junbin Gao. On Riemannian optimization over positive definite matrices with the Bures-Wasserstein geometry. *NeurIPS*, 2021.
- [93] Andi Han, Bamdev Mishra, Pratik Jawanpuria, and Junbin Gao. Learning with symmetric positive definite matrices via generalized Bures-Wasserstein geometry. In *International Conference on Geometric Science of Information*, pages 405–415. Springer, 2023.
- [94] Mehrtash Harandi, Mathieu Salzmann, and Richard Hartley. Dimensionality reduction on SPD manifolds: The emergence of geometry-aware methods. *IEEE TPAMI*, 2018.
- [95] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *IJCV*, 2013.
- [96] Doron Haviv, Aram-Alexandre Pooladian, Dana Pe’Er, and Brandon Amos. Wasserstein flow matching: Generative modeling over families of distributions. In *ICML*, 2025.
- [97] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [98] Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassioulas, Menglin Yang, and Rex Ying. HELM: Hyperbolic large language models via mixture-of-curvature experts. In *NeurIPS*, 2025.
- [99] Neil He, Menglin Yang, and Rex Ying. Lorentzian residual neural networks. In *KDD*, 2025.
- [100] Uwe Helmke and John B Moore. *Optimization and Dynamical Systems*. Springer Science & Business Media, 2012.
- [101] Marcel F. Hinss, Ludovic Darnet, Bertille Somon, Emilie Jahanpour, Fabien Lotte, Simon Ladouce, and Raphaëlle N. Roy. An EEG dataset for cross-session mental workload estimation: Passive BCI competition of the Neuroergonomics Conference 2021, 2021.
- [102] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

- [103] Chen Hu, Rui Wang, Xiaoning Song, Tao Zhou, Xiao-Jun Wu, Nicu Sebe, and Ziheng Chen. A correlation manifold self-attention network for EEG decoding. In *IJCAI*, 2025.
- [104] Chen Hu, Ziheng Chen, Rui Wang, Yefeng Zheng, and Nicu Sebe. Riemannian high-order pooling for brain foundation models. In *ICLR*, 2026.
- [105] Xiaoqiang Hua, Yongqiang Cheng, Hongqiang Wang, Yuliang Qin, Yubo Li, and Wenpeng Zhang. Matrix CFAR detectors based on symmetrized Kullback–Leibler and total Kullback–Leibler divergences. *Digital Signal Processing*, 69:106–116, 2017. URL <https://doi.org/10.1016/j.dsp.2017.06.019>.
- [106] Zhiwu Huang and Luc Van Gool. A Riemannian network for SPD matrix learning. In *AAAI*, 2017.
- [107] Zhiwu Huang, Chengde Wan, Thomas Probst, and Luc Van Gool. Deep learning on Lie groups for skeleton-based action recognition. In *CVPR*, 2017.
- [108] Zhiwu Huang, Jiqing Wu, and Luc Van Gool. Building deep networks on Grassmann manifolds. In *AAAI*, 2018.
- [109] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.
- [110] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Matrix backpropagation for deep networks with structured layers. In *ICCV*, 2015.
- [111] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Training deep networks with structured layers by matrix backpropagation. *arXiv preprint arXiv:1509.07838*, 2015.
- [112] Vinay Jayaram and Alexandre Barachant. MOABB: trustworthy algorithm benchmarking for BCIs. *Journal of Neural Engineering*, 15(6):066011, 2018.
- [113] Shaocheng Jin, Tao Zhou, Rui Wang, Ziheng Chen, Xiaoqing Luo, Xiao-Jun Wu, and Josef Kittler. Towards robust EEG decoding based on Riemannian self-attention. In *KDD*, 2026.
- [114] Dimitris Kalatzis, David Eklund, Georgios Arvanitidis, and Søren Hauberg. Variational autoencoders with riemannian brownian motion priors. In *ICML*, 2020.

-
- [115] Huan Kang, Hui Li, Tianyang Xu, Xiao-Jun Wu, Rui Wang, Chunyang Cheng, and Josef Kittler. SMLNet: A SPD manifold learning network for infrared and visible image fusion. *IJCV*, 2025.
- [116] Hermann Karcher. Riemannian center of mass and mollifier smoothing. *Communications on Pure and Applied Mathematics*, 30(5):509–541, 1977. URL <https://doi.org/10.1002/cpa.3160300502>.
- [117] Isay Katsman, Eric Ming Chen, Sidhanth Holalkere, Anna Asch, Aaron Lou, Ser-Nam Lim, and Christopher De Sa. Riemannian residual neural networks. In *NeurIPS*, 2023.
- [118] Isay Katsman, Eric Chen, Sidhanth Holalkere, Anna Asch, Aaron Lou, Ser Nam Lim, and Christopher M De Sa. Riemannian residual neural networks. *NeurIPS*, 2024.
- [119] Raiyan R Khan, Philippe Chlenski, and Itsik Pe’er. Hyperbolic genome embeddings. In *ICLR*, 2025.
- [120] Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In *CVPR*, 2020.
- [121] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [122] Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.
- [123] Reinmar Kobler, Jun-ichiro Hirayama, Qibin Zhao, and Motoaki Kawanabe. SPD domain-specific batch normalization to crack interpretable unsupervised domain adaptation in EEG. In *NeurIPS*, 2022.
- [124] Reinmar J Kobler, Jun-ichiro Hirayama, and Motoaki Kawanabe. Controlling the Fréchet variance improves batch normalization on the symmetric positive definite manifold. In *ICASSP*, 2022.
- [125] Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch. *arXiv preprint arXiv:2005.02819*, 2020.
- [126] A Krizhevsky. Learning multiple layers of features from tiny images. *Master’s thesis, University of Tront*, 2009.

- [127] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.
- [128] Serge Lang. *Algebra*. Springer Science & Business Media, 2012.
- [129] Yann Le and Xuan Yang. Tiny ImageNet visual recognition challenge, 2015.
- [130] Guy Lebanon and John Lafferty. Hyperplane margin classifiers on the multinomial manifold. In *ICML*, 2004.
- [131] John M Lee. *Introduction to Riemannian Manifolds*, volume 2. Springer, 2018.
- [132] Mario Lezcano Casado. Trivializations for gradient-based optimization on manifolds. In *NeurIPS*, 2019.
- [133] Peihua Li, Jiangtao Xie, Qilong Wang, and Zilin Gao. Towards faster training of global covariance pooling networks by iterative matrix square root normalization. In *CVPR*, 2018.
- [134] Shanglin Li, Motoaki Kawanabe, and Reinmar J Kobler. SPDIM: Source-free unsupervised conditional and label shift adaptation in EEG. In *ICLR*, 2025.
- [135] Shanglin Li, Shiwen Chu, Okan Koç, Yi Ding, Qibin Zhao, Motoaki Kawanabe, and Ziheng Chen. HEEGNet: Hyperbolic embeddings for EEG. *ICLR*, 2026.
- [136] Yancong Li, Xiaoming Zhang, Ying Cui, and Shuai Ma. Hyperbolic graph neural network for temporal knowledge graph completion. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 2024.
- [137] Zhenhua Lin. Riemannian geometry of symmetric positive definite matrices via Cholesky decomposition. *SIAM Journal on Matrix Analysis and Applications*, 40(4):1353–1370, 2019.
- [138] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding. *IEEE T-PAMI*, 2019.
- [139] Qi Liu, Maximilian Nickel, and Douwe Kiela. Hyperbolic graph neural networks. In *NeurIPS*, 2019.

-
- [140] Yuanpei Liu, Zhenqi He, and Kai Han. Hyperbolic category discovery. In *CVPR*, 2025.
- [141] Federico López, Beatrice Pozzetti, Steve Trettel, Michael Strube, and Anna Wienhard. Vector-valued distance and Gyrocalculus on the space of symmetric positive definite matrices. In *NeurIPS*, 2021.
- [142] Aaron Lou, Isay Katsman, Qingxuan Jiang, Serge Belongie, Ser-Nam Lim, and Christopher De Sa. Differentiating through the Fréchet mean. In *ICML*, 2020.
- [143] Miroslav Lovrić, Maung Min-Oo, and Ernst A Ruh. Multivariate normal distributions parametrized as a riemannian symmetric space. *Journal of Multivariate Analysis*, 74(1):36–48, 2000.
- [144] Jan R Magnus and Heinz Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 2019. URL <https://doi.org/10.1002/9781119541219>.
- [145] Luigi Malagò, Luigi Montrucchio, and Giovanni Pistone. Wasserstein Riemannian geometry of gaussian densities. *Information Geometry*, 1:137–179, 2018.
- [146] Jonathan H Manton. A globally convergent numerical algorithm for computing the centre of mass on compact Lie groups. In *The 8th Control, Automation, Robotics and Vision Conference, 2004.*, volume 3, pages 2211–2216. IEEE, 2004.
- [147] Yidan Mao, Jing Gu, Marcus C Werner, and Dongmian Zou. Klein model for hyperbolic neural networks. *arXiv preprint arXiv:2410.16813*, 2024.
- [148] Emile Mathieu and Maximilian Nickel. Riemannian continuous normalizing flows. In *NeurIPS*, 2020.
- [149] Debin Meng, Xiaojiang Peng, Kai Wang, and Yu Qiao. Frame attention networks for facial expression recognition in videos. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3866–3870. IEEE, 2019. URL <https://doi.org/10.1109/ICIP.2019.8803603>.
- [150] Aaron Meurer, Christopher P Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K Moore, Sartaj Singh, et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, 2017.

- [151] Hà Quang Minh. Alpha Procrustes metrics between positive definite operators: a unifying formulation for the Bures-Wasserstein and Log-Euclidean/Log-Hilbert-Schmidt metrics. *Linear Algebra and its Applications*, 636:25–68, 2022.
- [152] Maher Moakher. On the averaging of symmetric positive-definite tensors. *Journal of Elasticity*, 82(3):273–296, 2006. URL <https://doi.org/10.1007/s10659-005-9035-z>.
- [153] Meinard Müller, Tido Röder, Michael Clausen, Bernhard Eberhardt, Björn Krüger, and Andreas Weber. Documentation mocap database HDM05. Technical report, Universität Bonn, 2007.
- [154] Iain Murray. Differentiation of the Cholesky decomposition. *arXiv preprint arXiv:1602.07527*, 2016.
- [155] Galileo Namata, Ben London, Lise Getoor, Bert Huang, and U Edu. Query-driven active surveying for collective classification. In *10th International Workshop on Mining and Learning with Graphs*, 2012.
- [156] Xuan Son Nguyen. Geomnet: A neural network based on Riemannian geometries of SPD matrix space and Cholesky space for 3D skeleton-based interaction recognition. In *ICCV*, 2021.
- [157] Xuan Son Nguyen. The Gyro-structure of some matrix manifolds. In *NeurIPS*, 2022.
- [158] Xuan Son Nguyen. A Gyrovector space approach for symmetric positive semi-definite matrix learning. In *ECCV*, 2022.
- [159] Xuan Son Nguyen and Shuo Yang. Building neural networks on matrix manifolds: A Gyrovector space approach. In *ICML*, 2023.
- [160] Xuan Son Nguyen, Luc Brun, Olivier Lézoray, and Sébastien Boughleux. A neural network based on SPD manifold learning for skeleton-based hand gesture recognition. In *CVPR*, 2019.
- [161] Xuan Son Nguyen, Shuo Yang, and Aymeric Histace. Matrix manifold neural networks++. In *ICLR*, 2024.
- [162] Xuan Son Nguyen, Shuo Yang, and Aymeric Histace. Neural networks on symmetric spaces of noncompact type. In *ICLR*, 2025.

-
- [163] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *NeurIPS*, 2017.
- [164] Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In *ICML*, 2018.
- [165] Barrett O’Neill. *Semi-Riemannian Geometry with Applications to Relativity*, volume 103 of *Pure and Applied Mathematics*. Academic Press, New York, 1983.
- [166] Avik Pal, Max van Spengler, Guido Maria D’Amely di Melendugno, Alessandro Flaborea, Fabio Galasso, and Pascal Mettes. Compositional entailment learning for hyperbolic vision-language models. In *ICLR*, 2025.
- [167] Yue-Ting Pan, Jing-Lun Chou, and Chun-Shu Wei. MAtt: A manifold attention network for EEG decoding. In *NeurIPS*, 2022.
- [168] Yong-Hyun Park, Mingi Kwon, Jaewoong Choi, Junghyo Jo, and Youngjung Uh. Understanding the latent space of diffusion models through the lens of Riemannian geometry. In *NeurIPS*, 2023.
- [169] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019.
- [170] Xavier Pennec. *Probabilities and statistics on Riemannian manifolds: A geometric approach*. PhD thesis, INRIA, 2004.
- [171] Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A riemannian framework for tensor computing. *IJCV*, 2006.
- [172] Can Pouliquen, Mathurin Massias, and Titouan Vayer. Schur’s positive-definite network: Deep learning in the SPD cone with structure. In *ICLR*, 2025.
- [173] John G Ratcliffe. *Foundations of Hyperbolic Manifolds*. Springer, 2006.
- [174] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3D deep learning with PyTorch3D. *arXiv preprint arXiv:2007.08501*, 2020.
- [175] Herbert Robbins and Sutton Monroe. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

- [176] Salem Said, Lionel Bombrun, Yannick Berthoumieu, and Jonathan H Manton. Riemannian Gaussian distributions on the space of symmetric positive definite matrices. *IEEE TIT*, 2017.
- [177] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.
- [178] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. NTU RGB+ D: A large scale dataset for 3D human activity analysis. In *CVPR*, 2016.
- [179] Xianglong Shi, Ziheng Chen, Yunhan Jiang, and Nicu Sebe. Intrinsic Lorentz neural network. In *ICLR*, 2026.
- [180] Ryohei Shimizu, Yusuke Mukuta, and Tatsuya Harada. Hyperbolic neural networks++. In *ICLR*, 2021.
- [181] Ryohei Shimizu, YUSUKE Mukuta, and Tatsuya Harada. Hyperbolic neural networks++. In *ICLR*, 2021.
- [182] Ondrej Skopek, Octavian-Eugen Ganea, and Gary Bécigneul. Mixed-curvature variational autoencoders. In *ICLR*, 2020.
- [183] Yue Song, Nicu Sebe, and Wei Wang. Why approximate matrix square root outperforms accurate svd in global covariance pooling? In *ICCV*, 2021.
- [184] Yue Song, Nicu Sebe, and Wei Wang. On the eigenvalues of global covariance pooling for fine-grained visual recognition. *IEEE TPAMI*, 2022.
- [185] Yue Song, Nicu Sebe, and Wei Wang. Fast differentiable matrix square root. In *ICLR*, 2022.
- [186] Suvrit Sra and Reshad Hosseini. Conic geometric optimization on the manifold of positive definite matrices. *SIAM Journal on Optimization*, 25(1):713–739, 2015. URL <https://doi.org/10.1137/140978168>.
- [187] Shlomo Sternberg. *Lectures on differential geometry*, volume 316. American Mathematical Soc., 1999. URL <https://www.ams.org/journals/bull/1965-71-02/S0002-9904-1965-11286-1/S0002-9904-1965-11286-1.pdf>.

-
- [188] Yadong Sun, Xiaofeng Cao, Yu Wang, Wei Ye, Jingcai Guo, and Qing Guo. Geometry awakening: Cross-geometry learning exhibits superiority over individual structures. In *NeurIPS*, 2024.
- [189] Tanuj Sur, Samrat Mukherjee, Kaizer Rahaman, Subhasis Chaudhuri, Muhammad Haris Khan, and Biplab Banerjee. Hyperbolic uncertainty-aware few-shot incremental point cloud segmentation. In *CVPR*, 2025.
- [190] Yann Thanwerdas. *Riemannian and stratified geometries on covariance and correlation matrices*. PhD thesis, Université Côte d’Azur, 2022.
- [191] Yann Thanwerdas. Permutation-invariant log-Euclidean geometries on full-rank correlation matrices. *SIAM Journal on Matrix Analysis and Applications*, 45(2): 930–953, 2024. doi: 10.1137/22M1538144.
- [192] Yann Thanwerdas and Xavier Pennec. Is affine-invariance well defined on SPD matrices? a principled continuum of metrics. In *Geometric Science of Information: 4th International Conference, GSI 2019, Toulouse, France, August 27–29, 2019, Proceedings 4*, pages 502–510. Springer, 2019.
- [193] Yann Thanwerdas and Xavier Pennec. Exploration of balanced metrics on symmetric positive definite matrices. In *Geometric Science of Information: 4th International Conference, GSI 2019, Toulouse, France, August 27–29, 2019, Proceedings 4*, pages 484–493. Springer, 2019.
- [194] Yann Thanwerdas and Xavier Pennec. The geometry of mixed-Euclidean metrics on symmetric positive definite matrices. *Differential Geometry and its Applications*, 81:101867, 2022.
- [195] Yann Thanwerdas and Xavier Pennec. Theoretically and computationally convenient geometries on full-rank correlation matrices. *SIAM Journal on Matrix Analysis and Applications*, 43(4):1851–1872, 2022. doi: 10.1137/22M1471729.
- [196] Yann Thanwerdas and Xavier Pennec. O (n)-invariant Riemannian metrics on SPD matrices. *Linear Algebra and its Applications*, 661:163–201, 2023.
- [197] Loring W. Tu. *An introduction to manifolds*. Springer, 2011.
- [198] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: the missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

- [199] Abraham A Ungar. *Analytic hyperbolic geometry: Mathematical foundations and applications*. World Scientific, 2005.
- [200] Abraham Albert Ungar. *Analytic Hyperbolic Geometry and Albert Einstein’s Special Theory of Relativity (Second Edition)*. World Scientific, 2022.
- [201] Max Van Spengler, Erwin Berkhout, and Pascal Mettes. Poincaré ResNet. In *ICCV*, 2023.
- [202] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.
- [203] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3D skeletons as points in a Lie group. In *CVPR*, 2014.
- [204] Haoqi Wang, Zhizhong Li, and Wayne Zhang. Get the best of both worlds: Improving accuracy and transferability by Grassmann class representation. In *ICCV*, 2023.
- [205] Qilong Wang, Jiangtao Xie, Wangmeng Zuo, Lei Zhang, and Peihua Li. Deep cnns meet global covariance pooling: Better representation and generalization. *IEEE TPAMI*, 2020.
- [206] Qilong Wang, Zhaolin Zhang, Mingze Gao, Jiangtao Xie, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Towards a deeper understanding of global covariance pooling in deep learning: An optimization perspective. *IEEE TPAMI*, 2023.
- [207] Rui Wang, Xiao-Jun Wu, and Josef Kittler. SymNet: A simple symmetric positive definite manifold deep learning method for image set classification. *IEEE TNNLS*, 2021.
- [208] Rui Wang, Xiao-Jun Wu, Ziheng Chen, Tianyang Xu, and Josef Kittler. DreamNet: A deep Riemannian manifold network for SPD matrix learning. In *ACCV*, 2022.
- [209] Rui Wang, Xiao-Jun Wu, Ziheng Chen, Tianyang Xu, and Josef Kittler. Learning a discriminative SPD manifold neural network for image set classification. *Neural Networks*, 151:94–110, 2022.

-
- [210] Rui Wang, Chen Hu, Ziheng Chen, Xiao-Jun Wu, and Xiaoning Song. A Grassmannian manifold self-attention network for signal classification. In *IJCAI*, 2024.
- [211] Rui Wang, Xiao-Jun Wu, Ziheng Chen, Cong Hu, and Josef Kittler. SPD manifold deep metric learning for image set classification. *IEEE TNNLS*, 2024.
- [212] Rui Wang, Chen Hu, Xiaoning Song, Xiao-Jun Wu, Nicu Sebe, and Ziheng Chen. Towards a general attention framework on gyrovector spaces for matrix manifolds. In *NeurIPS*, 2025.
- [213] Rui Wang, Jiayao Jin, Ziheng Chen, Cong Wu, Xiao-Jun Wu, and Nicu Sebe. Structural topology refinement network for skeleton-based action recognition. *IEEE TIM*, 2025.
- [214] Rui Wang, Shaocheng Jin, Zhenyu Cai, Ziheng Chen, Xiao-Jun Wu, and Josef Kittler. Learning a better SPD network for signal classification: A Riemannian batch normalization method. *IEEE TNNLS*, 2025.
- [215] Rui Wang, Shaocheng Jin, Ziheng Chen, Xiaoqing Luo, and Xiao-Jun Wu. Learning to normalize on the SPD manifold under Bures-Wasserstein geometry. In *CVPR*, 2025.
- [216] Rui Wang, Zihao Bi, Chen Hu, Xiaoning Song, Xiao-Jun Wu, Nicu Sebe, and Ziheng Chen. Riemannian graph convolutional network for skeleton-based two-person interaction recognition. In *IJCAI*, 2026.
- [217] Rui Wang, Yuting Jiang, Xiaoqing Luo, Xiao-Jun Wu, Nicu Sebe, and Ziheng Chen. Wasserstein-aligned hyperbolic multi-view clustering. *AAAI*, 2026.
- [218] Yunfeng Wang and Gregory S Chirikjian. Error propagation on the euclidean group with applications to manipulator kinematics. *IEEE Transactions on Robotics*, 22(4):591–602, 2006.
- [219] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018.
- [220] Or Yair, Mirela Ben-Chen, and Ronen Talmon. Parallel transport on the cone manifold of SPD matrices for domain adaptation. *IEEE TIP*, 2019.
- [221] Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. Hypformer: Exploring efficient transformer fully in hyperbolic space. In *KDD*, 2024.

- [222] Menglin Yang, Ram Samarth B B, Aosong Feng, Bo Xiong, Jiahong Liu, Irwin King, and Rex Ying. Hyperbolic fine-tuning for large language models. In *NeurIPS*, 2025.
- [223] Xin Yang, Xingrun Li, Heng Chang, Xihong Yang, Shengyu Tao, Maiko Shigeno, Ningkang Chang, Junfeng Wang, Dawei Yin, Erxue Min, et al. Hgformer: Hyperbolic graph transformer for collaborative filtering. In *ICML*, 2025.
- [224] Ryoma Yataka, Kazuki Hirashima, and Masashi Shiraishi. Grassmann manifold flows for stable shape generation. In *NeurIPS*, 2023.
- [225] Florian Yger. A review of kernels on covariance matrices for BCI applications. In *2013 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2013. URL <https://doi.org/10.1109/MLSP.2013.6661972>.
- [226] Hongwei Yong, Jianqiang Huang, Deyu Meng, Xiansheng Hua, and Lei Zhang. Momentum batch normalization for deep learning with small batch size. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*, pages 224–240. Springer, 2020.
- [227] Ying Yuan, Hongtu Zhu, Weili Lin, and James Stephen Marron. Local polynomial regression for symmetric positive definite matrices. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 74(4):697–719, 2012.
- [228] Ernesto Zacur, Matias Bossa, and Salvador Olmos. Left-invariant Riemannian geodesics on spatial transformation groups. *SIMAX*, 2014.
- [229] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *NeurIPS*, 2018.
- [230] Tong Zhang, Wenming Zheng, Zhen Cui, Yuan Zong, Chaolong Li, Xiaoyan Zhou, and Jian Yang. Deep manifold-to-manifold transforming network for skeleton-based action recognition. *IEEE TMM*, 2020.
- [231] Wei Zhao, Federico Lopez, J Maxwell Riestenberg, Michael Strube, Diaaeldin Taha, and Steve Trettel. Modeling graphs beyond hyperbolic: Graph neural networks in symmetric positive definite matrices. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 122–139. Springer, 2023.

- [232] Xingjian Zhen, Rudrasis Chakraborty, Nicholas Vogt, Barbara B Bendlin, and Vikas Singh. Dilated convolutional neural networks for sequential manifold-valued data. In *ICCV*, 2019.
- [233] Runhe Zhou, Shanglin Li, Guanxiang Huang, Xinliang Zhou, Qibin Zhao, Motoaki Kawanabe, Yi Ding, and Cuntai Guan. EEG-based multimodal learning via hyperbolic mixture-of-curvature experts. In *ICML*, 2026.
- [234] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. DNABERT-2: Efficient foundation model and benchmark for multi-species genome. In *ICLR*, 2024.

Appendix A

Experimental Details and Additional Discussions

A.1 Data Sets

This section collects descriptions of the data sets used in the experimental chapters.

A.1.1 Skeleton-Based Action Recognition and Gesture Data Sets

HDM05 [153]. The HDM05 data set¹ consists of 2,273 skeleton-based motion capture sequences executed by different actors. Each frame records the 3D coordinates of 31 joints. Different experiments use the standard task-specific filtered versions adopted by their corresponding backbones, as detailed in the method-specific experimental settings.

FPHA [80]. The FPHA data set² includes 1,175 skeleton-based first-person hand gesture videos of 45 different categories, with 600 clips for training and 575 for testing. Each frame contains the 3D coordinates of 21 hand joints.

G3D [23]. The G3D data set consists of 663 sequences of 20 different gaming actions. Each sequence records the 3D locations of 20 joints, *i.e.*, 19 bones.

NTU60 [178]. The NTU60 data set³ contains 56,880 skeleton sequences classified into 60 classes, where each frame includes 3D coordinates of 25 or 50 body joints.

NTU120 [138]. The NTU120 data set⁴ contains 114,480 sequences in 120 action classes.

¹<https://resources.mpi-inf.mpg.de/HDM05/>

²https://github.com/guiggh/hand_pose_action

³<https://github.com/shahrouty/NTURGB-D>

⁴<https://github.com/shahrouty/NTURGB-D>

Data Set	#Nodes	#Edges	#Classes	#Features
Disease	1044	1043	2	1000
Airport	3188	18631	4	4
PubMed	19717	44338	3	500
Cora	2708	5429	7	1433

Table A.1: Summary statistics for the graph data sets.

A.1.2 Radar and EEG Signal Data Sets

Radar [31]. The Radar data set⁵ contains 3,000 synthetic radar signals equally distributed in 3 classes.

Hinss2021 [101]. The Hinss2021 data set⁶ is a competition data set containing EEG signals for mental workload estimation. The data set is employed for two tasks, inter-session and inter-subject, which are treated as domain adaptation problems. Geometry-aware methods [220, 123] have demonstrated promising performance in EEG classification.

A.1.3 Image Classification Data Sets

AFEW [68]. The *Acted Facial Expressions in the Wild (AFEW)* data set contains seven emotion categories, with 773 videos for training and 383 for validation.

CIFAR-10 and CIFAR-100 [126]. The CIFAR-10 and CIFAR-100 data sets each contain 60,000 32×32 color images from 10 and 100 classes, respectively. We use the standard PyTorch splits: 50,000 training images and 10,000 test images.

Tiny-ImageNet [129]. Tiny-ImageNet is a subset of ImageNet with 100,000 images from 200 classes, resized to 64×64 . We use the official validation split for evaluation.

ImageNet-1k [66]. ImageNet-1k contains 1.28M training images, 50K validation images, and 100K test images distributed across 1k classes.

A.1.4 Graph Data Sets

Cora [177]. Cora is a citation network where the nodes represent scientific papers in the area of machine learning, the edges are citations between them, and the labels of the nodes are academic subareas.

⁵<https://www.dropbox.com/s/dfnlx2bnyh3kjwy/data.zip?dl=0>

⁶<https://zenodo.org/record/5055046>

Task	Species	Data Sets	Num. classes	Max length	Train / Dev / Test
Retrotransposons	Plant	LTR Copia	2	500	7666 / 682 / 568
		LINEs		1000	22502 / 2030 / 1782
		SINEs		500	21152 / 1836 / 1784
DNA transposons	Plant	CMC-EnSpm	2	200	19912 / 1872 / 1808
		hAT-Ac		1000	17322 / 1822 / 1428
Pseudogenes	Human	processed	2	1000	17956 / 1046 / 1740
		unprocessed			12938 / 766 / 884

Table A.2: Summary statistics for TEB.

Disease [4]. Disease represents a disease propagation tree that simulates the susceptible, infected, and recovered (SIR) disease transmission model, with each node representing either an infection or a non-infection state.

Airport [229]. Airport is a transductive data set where nodes represent airports and edges represent airline routes from OpenFlights.org.

PubMed [155]. PubMed is a standard benchmark that describes citation networks where nodes represent scientific papers in the area of medicine, the edges are citations between them, and the node labels are academic subareas.

Tab. A.1 summarizes the above data sets.

A.1.5 Genomic Sequence Data Sets

Transposable Elements Benchmark [119]. The *Transposable Elements Benchmark (TEB)* comprises seven binary classification data sets that investigate transposable elements across plant and human genomes. The seven data sets are LTR Copia, LINEs, and SINEs for plant retrotransposons; CMC-EnSpm and hAT-Ac for plant DNA transposons; and processed and unprocessed pseudogenes for human pseudogenes. For each data set, positive examples are sequences spanning annotated elements of interest, and negatives are randomly sampled, non-overlapping genomic segments outside these regions. We adopt chromosome-level training, validation, and test splits, using chromosomes 8 and 9 for validation and test in plant genomes, and chromosomes 20–22 and 17–19 for validation and test in human genomes, respectively. Tab. A.2 provides the summary statistics.

Genome Understanding Evaluation [234]. The *Genome Understanding Evaluation (GUE)* benchmark contains seven biologically significant genome analysis tasks that span 28 data sets. The data sets contain sequences ranging from 70 to 1000 base pairs in length and originate from yeast, mouse, human, and virus genomes. The HBNN experiments select Core promoter detection and Promoter detection, together with the

A.2. Backbone Networks

Task	Species	Data Set	Num. classes	Length	Train / Dev / Test
Core promoter detection	Human	tata	2	70	4904 / 613 / 613
		notata			42452 / 5307 / 5307
		all			47356 / 5920 / 5920
Promoter detection	Human	tata	2	300	4904 / 613 / 613
		notata			42452 / 5307 / 5307
		all			47356 / 5920 / 5920
Covid variant classification	Virus	Covid	9	1000	77669 / 7000 / 7000
Species classification	Fungi	Fungi	25	5000	8000 / 1000 / 1000
	Virus	Virus	20	10000	4000 / 500 / 500

Table A.3: Summary statistics for the adopted GUE data sets.

multi-class tasks Covid variant classification and species classification, totaling nine data sets. Tab. A.3 provides their summary statistics.

A.2 Backbone Networks

This section collects the backbone network architectures used in the thesis.

A.2.1 Backbone Networks on the SPD Manifold

SPDNet. SPDNet [106] is a classical SPD neural network. It mimics conventional densely connected feedforward networks and consists of three basic building blocks:

$$\text{BiMap layer: } S^k = W^k S^{k-1} W^{k\top}, \text{ with } W^k \text{ semi-orthogonal,} \quad (\text{A.1})$$

$$\text{ReEig layer: } S^k = U^{k-1} \max(\Sigma^{k-1}, \epsilon I_n) U^{k-1\top}, \text{ with } S^{k-1} = U^{k-1} \Sigma^{k-1} U^{k-1\top}, \quad (\text{A.2})$$

$$\text{LogEig layer: } S^k = \log(S^{k-1}). \quad (\text{A.3})$$

where $\max(\cdot)$ is element-wise maximization. BiMap and ReEig mimic transformation and non-linear activation, while LogEig maps SPD matrices into the tangent space at the identity matrix for classification.

SPDNetBN. SPDNetBN [31] further proposed RBN based on AIM:

$$\text{Centering from mean } M \in \mathcal{S}_{++}^n : \forall i \leq N, \bar{P}_i \leftarrow M^{-\frac{1}{2}} P_i M^{-\frac{1}{2}}, \quad (\text{A.4})$$

$$\text{Biasing towards parameter } B \in \mathcal{S}_{++}^n : \forall i \leq N, \hat{P}_i \leftarrow B^{\frac{1}{2}} \bar{P}_i B^{\frac{1}{2}}. \quad (\text{A.5})$$

TSMNet and SPDDSMBN. TSMNet [123] can be illustrated as $f_{tc} \rightarrow f_{sc} \rightarrow f_{BiMap} \rightarrow f_{ReEig} \rightarrow f_{LogEig}$, where f_{tc} and f_{sc} denote temporal and spatial convolution,

respectively. SPDDSMBN [123] is an improved version of SPDNetBN. Apart from controlling the mean, it can also control variance. The key operation in SPDDSMBN for controlling the mean and variance is

$$\forall i \leq N, \quad \bar{P}_i \leftarrow \Gamma_{I \rightarrow B}[(\Gamma_{M \rightarrow I}(P_i))^{\frac{s}{v}}], \quad (\text{A.6})$$

where M is the Riemannian mean, v^2 is the Fréchet variance, $B \in \mathcal{S}_{++}^n$ is the biasing parameter, and $s \in \mathbb{R}$ is the scaling factor. Inspired by Yong et al. [226], during training, SPDDSMBN generates running means and running variances for training and testing with distinct momentum parameters. It uses the training running statistics during training and the testing running statistics during testing. SPDDSMBN also applies domain-specific techniques [40], keeping multiple parallel BN layers and distributing observations according to the associated domains. To share cross-domain knowledge, s is uniformly learned across all domains, and B is set to the identity matrix. Kobler et al. [123] adopted SPDDSMBN for domain adaptation in EEG classification.

SPDGCN [231]. SPDGCN is used as a Riemannian graph neural network backbone.

GyroSPD [159]. GyroSPD substitutes the BiMap layer in SPDNet with the AIM-based gyrotranslation $S^k = W^k \oplus^{\text{AI}} S^{k-1} = (W^k)^{\frac{1}{2}} S^{k-1} (W^k)^{\frac{1}{2}}$, where W^k is an SPD matrix parameter.

A.2.2 Backbone Networks on the Grassmannian

GyroGr. GyroGr [159] mimics conventional densely connected feedforward networks and is composed of three basic building blocks. Given an ONB Grassmannian matrix U^{k-1} , the gyrotranslation and ProjMap layers are defined as

$$\text{Gyrotranslation: } U^k = W^k \oplus^{\text{Gr}} U^{k-1}, \quad W^k \in \text{Gr}(p, n), \quad (\text{A.7})$$

$$\text{ProjMap: } P^k = U^{k-1}(U^{k-1})^\top. \quad (\text{A.8})$$

In addition, the pooling is performed via the space of projection matrices [108]. Specifically, Grassmannian data are first mapped to the space of projection matrices via the ProjMap layer. A standard mean pooling operation is applied to the resulting projection matrices. Finally, the pooled matrices are projected back to the ONB

Grassmannian by SVD. The entire procedure can be expressed as

$$\begin{aligned} P^k &= f_p \left(U^{k-1} (U^{k-1})^\top \right), \\ U^k &= O_{1:p}^k, \quad \text{where } P^k \stackrel{\text{SVD}}{:=} O^k \Sigma^k (O^k)^\top, \end{aligned} \tag{A.9}$$

where f_p is a regular mean pooling.

A.2.3 Backbone Networks on Rotation Matrices

LieNet. LieNet [107] is a classical neural network on rotation matrices. Its latent space is the Lie group $\text{SO}^N(3) = \text{SO}(3) \times \cdots \times \text{SO}(3)$, *i.e.*, $R = (R_1, \dots, R_N) \in \text{SO}^N(3)$. The group and manifold structures on $\text{SO}^N(3)$ are defined component-wise. For instance, $R^1 \odot R^2 = (R_1^1 R_1^2, \dots, R_N^1 R_N^2)$. There are three basic layers in LieNet:

$$\text{RotMap layer: } R^k = W^k \odot R^{k-1}, \text{ with } W^k \in \text{SO}^N(3), \tag{A.10}$$

$$\text{RotPooling layer: } R_i^k = \begin{cases} R_{m_i, n_i}^{k-1}, & \text{if } \Theta(R_{m_i, n_i}^{k-1}) > \Theta(R_{n_i, m_i}^{k-1}), \\ R_{n_i, m_i}^{k-1}, & \text{otherwise,} \end{cases} \tag{A.11}$$

$$\text{LogMap layer: } R^k = \log(R^{k-1}), \tag{A.12}$$

where $\Theta(\cdot)$ is the Euler angle, and (n_i, m_i) are two indices. The RotMap and RotPooling layers mimic the convolution and pooling layers, while the LogMap layer maps rotation matrices into the tangent space for classification.

In LieNet, each rotation feature has shape $[\text{num}, \text{frame}, 3, 3]$, where num and frame denote the spatial and temporal dimensions. The RotPooling layer is applied along either the spatial or temporal dimension, while the RotMap layer is applied along the spatial dimension, with W^k of size $[\text{num}, 3, 3]$.

A.2.4 Backbone Networks on Hyperbolic Spaces

We briefly recap the hyperboloid layers, Poincaré MLR, Poincaré FC layer, and Poincaré β -concatenation used in the experiments. Throughout this subsection, $K < 0$.

Hyperboloid Neural Networks. Let $x \in \mathbb{H}_K^n$ be the input vector. The weight parameters are $W \in \mathbb{R}^{m \times (n+1)}$ and $v \in \mathbb{R}^{n+1}$. The Lorentz FC layer [45, Eq. 3] and

activation layer [15, Eq. 13] are defined in a spacetime manner:

$$\text{Activation: } y = \begin{bmatrix} \sqrt{\|\psi(x_s)\|^2 - 1/K} \\ \psi(x_s) \end{bmatrix}, \quad (\text{A.13})$$

$$\text{FC: } y = \begin{bmatrix} \sqrt{\|\phi(Wx, v)\|^2 - 1/K} \\ \phi(Wx, v) \end{bmatrix}, \quad (\text{A.14})$$

$$\text{with } \phi(Wx, v) = \lambda \sigma(v^\top x + b') \frac{W\psi(x) + b}{\|W\psi(x) + b\|}, \quad (\text{A.15})$$

where $b \in \mathbb{R}^m$ and $b' \in \mathbb{R}$ are biases, ψ is an activation function, σ is the sigmoid function, and $\lambda > 0$ is a learnable scaling parameter.

Poincaré MLR. Lebanon and Lafferty [130] first reformulated the Euclidean MLR $p(y=k | x) \propto \exp(\langle a_k, x \rangle - b_k)$ via the point-to-hyperplane distance:

$$p(y=k | x) \propto \exp(\text{sign}(\langle a_k, x \rangle - b_k) \|a_k\| d(x, H_{a_k, b_k})), \quad (\text{A.16})$$

$$H_{a, b} = \{x \in \mathbb{R}^n \mid \langle a, x \rangle - b = 0\}, \quad \text{where } a \in \mathbb{R}^n \setminus \{\mathbf{0}\} \text{ and } b \in \mathbb{R}. \quad (\text{A.17})$$

For $x \in \mathbb{P}_K^n$, Ganea et al. [76, Eqs. 24–25] generalized this formulation via geometric reinterpretation, and Shimizu et al. [181, Sec. 3.1] further derived the resulting closed form:

$$v_k(x) = \frac{2\|z_k\|}{\sqrt{|K|}} \text{asinh} \left(\lambda_x^K \langle \sqrt{|K|}x, [z_k] \rangle \cosh(2\sqrt{|K|}r_k) - (\lambda_x^K - 1) \sinh(2\sqrt{|K|}r_k) \right),$$

where $\lambda_x^K = 2(1 - |K|\|x\|^2)^{-1}$ is the conformal factor, $p(y=k | x) \propto \exp(v_k(x))$, and $[z_k] = \frac{z_k}{\|z_k\|}$. Here, $z_k \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ and $r_k \in \mathbb{R}$ are parameters. Under the identification $a_k = z_k$ and $b_k = \|z_k\| r_k$, the Euclidean limit is $\lim_{K \rightarrow 0} v_k(x) = 4(\langle a_k, x \rangle - b_k)$.

Poincaré FC Layer. Shimizu et al. [181] extended the Euclidean FC layer to the Poincaré ball via point-to-hyperplane distances. In Euclidean spaces, the FC can be written element-wise as $y_k = \langle a_k, x \rangle - b_k$ with $x, a_k \in \mathbb{R}^n$ and $b_k \in \mathbb{R}$. Thus, y_k equals $\|a_k\|$ times the signed distance from x to the hyperplane H_{a_k, b_k} . Under this interpretation, the Poincaré FC layer $\mathcal{F} : \mathbb{P}_K^n \rightarrow \mathbb{P}_K^m$ takes the closed form

$$y = \frac{w}{1 + \sqrt{1 + |K|\|w\|^2}}, \quad w_k = |K|^{-1/2} \sinh \left(\sqrt{|K|} v_k(x) \right), \quad (\text{A.18})$$

where $|K|$ is the magnitude of the curvature. Here, $Z = \{z_k\}_{k=1}^m$ and $r = \{r_k\}_{k=1}^m$ parameterize the orientations and biases, and $v_k(x)$ is the Poincaré MLR logit.

Poincaré β -Concatenation. It generalizes the Euclidean concatenation into the hyperbolic Poincaré ball, stabilizing the norm of the Poincaré vector [181, Sec. 3.3]. Given inputs $\{x_i \in \mathbb{P}_K^{n_i}\}_{i=1}^N$, it is defined as

$$\text{Exp}_0 \left(\beta_n \left(\beta_{n_1}^{-1} v_1^\top, \dots, \beta_{n_N}^{-1} v_N^\top \right) \right)^\top \in \mathbb{P}_K^n, \quad (\text{A.19})$$

where $v_i = \text{Log}_0(x_i)$, $n = \sum_{i=1}^N n_i$, and $\beta_\alpha = B(\alpha/2, 1/2)$ is defined using the beta function.

A.2.5 Riemannian Residual Network Backbones

RResNet. Euclidean residual blocks can be written as

$$x^{(i)} = x^{(i-1)} + n_i \left(x^{(i-1)} \right), \quad (\text{A.20})$$

where n_i is a network. Katsman et al. [118] generalized this to manifolds by replacing addition with the Riemannian exponential map:

$$x^{(i)} = \text{Exp}_{x^{(i-1)}} \left(\ell_i(x^{(i-1)}) \right), \quad (\text{A.21})$$

where $\ell_i : \mathcal{M} \rightarrow T\mathcal{M}$ outputs a vector field parameterized by the neural network. As $\text{Exp}_x(v) = x + v$ for the Euclidean space, it can be immediately shown that Eq. (A.21) naturally extends Eq. (A.20) to manifolds. On the SPD manifold, the vector field is generated from the eigenvalues. Specifically, the SPD residual block [118, Eqs. 22–23] is

$$Y = \text{Exp}_X \left(Q \text{diag} \left(f \left(\text{spec}(X) \right) \right) Q^\top \right), \quad (\text{A.22})$$

where $X \in \mathcal{S}_{++}^n$, $\text{spec}(X)$ contains all eigenvalues, $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a neural network, and Q is an orthogonal parameter. Here, $\text{diag}(\cdot)$ returns a diagonal matrix from the input vector.

A.3 Experimental Details

A.3.1 Lie Group Batch Normalization

A.3.1.1 LieBN on the SPD Manifold

The SPDNet and TSMNet backbones, including SPDNetBN and SPDDSMBN, are reviewed in Sec. A.2.1. We next describe the domain-specific momentum LieBN and implementation details.

SPD modeling and preprocessing. The data set descriptions are collected in Sec. A.1. Throughout this thesis, every sample covariance matrix is made strictly positive definite before subsequent manifold operations by adding a small diagonal perturbation, $\Sigma \leftarrow \Sigma + \epsilon I_n$ with $\epsilon > 0$. Unless otherwise stated, this preprocessing convention is applied to all covariance-based SPD representations. Following the protocol of Brooks et al. [31], each Radar signal is divided into windows of length 20, whose series yields one 20×20 SPD covariance matrix. This produces 3,000 covariance matrices equally distributed across 3 classes. For HDM05, each frame consists of 3D coordinates of 31 joints, and each sequence is modeled by a 93×93 covariance matrix. Following the protocol of Brooks et al. [31], we trim the data set down to 2,086 sequences scattered throughout 117 classes by removing some under-represented classes. For the FPFA, we follow Wang et al. [207] to represent each sequence as a 63×63 covariance matrix. For Hinss2021, we choose the SOTA method, TSMNet [123], as our baseline model. We follow the Python implementation⁷ of Kobler et al. [123] to carry out preprocessing. In detail, the Python packages MOABB [112] and MNE [84] are used to preprocess the data set. The applied steps include resampling the EEG signals to 250/256 Hz, applying temporal filters to extract oscillatory EEG activity in the 4–36 Hz range, extracting short segments ($\leq 3s$) associated with a class label, and finally obtaining 40×40 SPD covariance matrices.

Domain-specific momentum LieBN for EEG classification. Kobler et al. [123] proposed SPDDSMBN as a domain adaptation approach for EEG classification. SPDDSMBN, based on Eq. (3.6), performed normalization of mean and variance on SPD manifolds under the specific AIM. Additionally, SPDDSMBN utilized separate momentum parameters for updating training and testing running statistics, inspired by Yong et al. [226]. Following Kobler et al. [123, Alg. 1], we also present a momentum

⁷<https://github.com/rkobler/TSMNet>

Algorithm 3: Momentum LieBN (MLieBN) Algorithm

Input : A batch of activations $\{P_i\}_{i=1}^N$ over the Lie group $\{\mathcal{M}, \oplus, g\}$,
and a small positive constant ϵ
running mean $\bar{M}_r = E$, running variance $\bar{v}_r^2 = 1$ for training
running mean $\tilde{M}_r = E$, running variance $\tilde{v}_r^2 = 1$ for testing
biasing parameter $B \in \mathcal{M}$, scaling parameter $s \in \mathbb{R} \setminus \{0\}$,
momentum parameters for training and testing $\eta_{\text{train}}, \eta \in [0, 1]$

Output : Normalized activations $\{\tilde{P}_i\}_{i=1}^N$

if *training* **then**

- Compute batch mean M_b and variance v_b^2 of $\{P_i\}_{i=1}^N$;
- $\bar{M}_r \leftarrow \text{WFM}(\{1 - \eta_{\text{train}}, \eta_{\text{train}}\}, \{\bar{M}_r, M_b\})$;
- $\bar{v}_r^2 \leftarrow (1 - \eta_{\text{train}})\bar{v}_r^2 + \eta_{\text{train}}v_b^2$;
- $\tilde{M}_r \leftarrow \text{WFM}(\{1 - \eta, \eta\}, \{\tilde{M}_r, M_b\})$;
- $\tilde{v}_r^2 \leftarrow (1 - \eta)\tilde{v}_r^2 + \eta v_b^2$;

if *training* **then** $M \leftarrow \bar{M}_r, v^2 \leftarrow \bar{v}_r^2$;

else $M \leftarrow \tilde{M}_r, v^2 \leftarrow \tilde{v}_r^2$;

for $i \leftarrow 1$ **to** N **do**

- Centering to the neutral element E :
 - if** g is left-invariant **then** $\bar{P}_i \leftarrow L_{\ominus M}(P_i)$;
 - else** $\bar{P}_i \leftarrow R_{\ominus M}(P_i)$;
- Scaling the dispersion:
 - $\hat{P}_i \leftarrow \text{Exp}_E \left[\frac{s}{\sqrt{v^2 + \epsilon}} \text{Log}_E(\bar{P}_i) \right]$
- Biasing towards parameter B :
 - if** g is left-invariant **then** $\tilde{P}_i \leftarrow L_B(\hat{P}_i)$;
 - else** $\tilde{P}_i \leftarrow R_B(\hat{P}_i)$;

LieBN (MLieBN) in Alg. 3. Here η is fixed and η_{train} is defined as

$$\eta_{\text{train}} = 1 - \rho^{\frac{1}{T-1} \max(T-t, 0)} + \rho, \quad \text{where} \quad \rho = \frac{1}{\text{domains_per_batch}}, \quad (\text{A.23})$$

where T and t denote the total number of training epochs and the current epoch, respectively. Furthermore, following Kobler et al. [123], we adopt multi-channel mechanisms for domain-specific MLieBN (DSMLieBN), where each domain has its own MLieBN layer. Similar to Kobler et al. [123], we set the biasing parameter equal to the neutral element, and the scaling factor is shared across all domains. We denote Alg. 3 as $\text{MLieBN}(P_j \mid B, s, \epsilon, \eta, \eta_{\text{train}})$. Then our DSMLieBN follows

$$\text{DSMLieBN}(P_j, i) = \text{MLieBN}_i(P_j \mid E, s, \epsilon, \eta, \eta_{\text{train}}), \quad \forall P_j \in \{P_k\}_{k=1}^N, \quad (\text{A.24})$$

where i is the index of the domain. We follow the official code of SPDDSMBN⁸ to implement our DSMLieBN. Thus, the only difference between DSMLieBN and SPDDSMBN is how normalization is performed. Analogous to Thm. 72, computations for DSMLieBN under pullback metrics can also be performed by mapping, calculating, and then remapping.

Implementation details. We use the official code of SPDNetBN⁹ [31] and TSMNet¹⁰ [123] to implement our experiments on the SPDNet and TSMNet backbones. For the SPDNet architecture, we compare our LieBN with SPDNetBN [31], which applies the SPDBN (Eqs. (3.2) and (3.3)) to SPDNet. Similar to SPDNetBN, we apply our LieBN after each transformation layer. In the EEG application, one of the state-of-the-art methods is TSMNet+SPDDSMBN [123], which is a domain adaptation version of Kobler et al. [124]. For a fair comparison, we also implement a domain-specific momentum LieBN, referred to as DSMLieBN. Following Kobler et al. [123], we apply our DSMLieBN before the LogEig layer in TSMNet. We use the standard cross-entropy loss and optimize the parameters with the Riemannian AMSGrad optimizer [18]. The network architectures are represented as $\{d_0, d_1, \dots, d_L\}$, where the dimension of the parameter in the i -th BiMap layer is $d_i \times d_{i-1}$. The experiments are conducted with a learning rate of $5e^{-3}$, a batch size of 30, and 200 training epochs on the Radar, HDM05, and FPHA data sets. For the Hinss2021 data set, following Kobler et al. [123], we use a learning rate of $1e^{-3}$ with a weight decay of $1e^{-4}$, a batch size of 50, and 50 training epochs.

Scoring metrics. In line with the previous work [31, 123], we use accuracy as the scoring metric for the Radar, HDM05, and FPHA data sets, and balanced accuracy (*i.e.*, the average recall across classes) for the Hinss2021 data set. Ten-fold experiments on the Radar, HDM05, and FPHA data sets are carried out with randomized initialization and split (the split is officially fixed for the FPHA data set), while on the Hinss2021 data set, models are fit and evaluated with randomized leave-5%-of-sessions-out (inter-session) or leave-5%-of-subjects-out (inter-subject) cross-validation.

A.3.1.2 LieBN on Rotation Matrices

Data sets and preprocessing. The data set descriptions are collected in Sec. A.1. Following Huang et al. [107], the G3D experiments use the cross-subject setting, with

⁸<https://github.com/rkobler/TSMNet>

⁹https://proceedings.neurips.cc/paper_files/paper/2019/file/6e69ebbfad976d4637bb4b39de261bf7-Supplemental.zip

¹⁰<https://github.com/rkobler/TSMNet>

half of the subjects used for training and the other half for testing, while the NTU60 experiments use the cross-view protocol [178]. We use the code¹¹ of Vemulapalli et al. [203] to represent each skeleton sequence as a point on the Lie group $\text{SO}^{N \times T}(3)$, where N and T denote spatial and temporal dimensions. As preprocessed in Huang et al. [107], we set T to 100, 16, and 64 on the G3D, HDM05, and NTU60 data sets, respectively.

LieNet. The LieNet backbone is reviewed in Sec. A.2.3. Note that the official code of LieNet¹² is implemented in MATLAB. We use the open-source PyTorch code¹³ to implement our experiments. To reproduce LieNet more faithfully, we made the following modifications to this PyTorch code. We reimplemented the LogMap and RotPooling layers to make them consistent with the official MATLAB implementation. In addition, we extended the Riemannian computations of `Geoopt` [125] to $\text{SO}(3)$ to enable the direct Riemannian optimization, which is missing from the current package. We apply our LieBN before the LogMap layer. Note that the dimension of input features in LieNet is $B \times N \times T \times 3 \times 3$. We calculate Lie group statistics along the batch and temporal dimensions ($B \times T$). We denote the LieNet models with our LieBN-Left and LieBN-Right as LieNetLieBN-Left and LieNetLieBN-Right, respectively.

Implementation details. We find that SGD is the most effective optimizer for LieNet, and thus, we adopt it for our experiments. The learning rate is set to $1e^{-2}$. The batch sizes are 30, 30, and 256 for the G3D, HDM05, and NTU60 data sets, respectively. On the NTU60 data set, the learning rate is reduced by a factor of 10 upon model convergence, specifically at the 5th and 25th epochs for LieNetLieBN and LieNet, respectively. For each model, we apply `torch.nn.utils.clip_grad_norm_` with `max_norm=5` to the transformation matrix in the final FC layer.

A.3.1.3 LieBN on Correlation Matrices

We follow the same settings as the experiments on the SPD manifold with respect to the backbone architecture, batch size, number of training epochs, optimizer, and learning rate. The network architecture can be denoted as BiMap-[Power-Cov2Cor-LieBN-Cor]-LogEig, where Power denotes the matrix power and Cov2Cor is $\text{Cor}(\cdot) : \mathcal{S}_{++}^n \rightarrow \text{Cor}^+(n)$. The matrix powers used for each data set are presented in Tab. A.4. A single iteration is sufficient to achieve saturated network performance with respect to calculating $\mathcal{D}$ and $\mathcal{D}^*$ in OLM and LSM, except for $\mathcal{D}^*$ on the HDM05 data set, which requires convergence with a maximum of 20 iterations.

¹¹<https://ravitejav.weebly.com/kbac.html>

¹²<https://github.com/zhiwu-huang/LieNet>

¹³<https://github.com/hjf1997/LieNet>

Data Set \ Metric	ECM	LECM	OLM	LSM
HDM05	0.75	0.5	0.5	-0.5
FPHA	-0.5	-0.25	-0.25	-0.25

Table A.4: Matrix powers in LieBN-Cor under different metrics on each data set.

A.3.2 Gyrogroup Batch Normalization

A.3.2.1 GyroBN on the Grassmannian

For HDM05, under-represented clips are removed, yielding 2,086 instances over 117 classes. For NTU60 and NTU120, we focus on mutual actions and adopt the cross-view and cross-setup protocols, respectively [178, 138]. Following Nguyen and Yang [159], each sequence is represented as a Grassmannian matrix of size 93×10 , 150×10 , and 150×10 for HDM05, NTU60, and NTU120, respectively. Because GyroBN is inserted after the first pooling layer, the corresponding inputs to GyroBN have sizes 47×10 , 75×10 , and 75×10 , as reported in Tab. 3.16. The GyroGr backbone is reviewed in Sec. A.2.2.

Trivialization. Following Nguyen and Yang [159], we apply the trivialization strategy reviewed in Sec. 2.7 to the Grassmannian parameters in the gyrotranslation and GyroBN layers. Each Grassmannian parameter $U \in \text{Gr}(p, n)$ is parameterized by a matrix $\mathfrak{U} \in \mathbb{R}^{(n-p) \times p}$ such that

$$\begin{bmatrix} \mathbf{0} & -\mathfrak{U}^\top \\ \mathfrak{U} & \mathbf{0} \end{bmatrix} = \left[\overline{UU^\top}, \tilde{I}_{p,n} \right], \quad (\text{A.25})$$

where $\overline{(\cdot)} = \text{Log}_{\tilde{I}_{p,n}}(\cdot)$. The parameter U can be retrieved by

$$U = \exp \left(\left[\overline{UU^\top}, \tilde{I}_{p,n} \right] \right) I_{p,n} = \exp \left(\begin{bmatrix} \mathbf{0} & -\mathfrak{U}^\top \\ \mathfrak{U} & \mathbf{0} \end{bmatrix} \right) I_{p,n}. \quad (\text{A.26})$$

This reparameterization represents the trainable Grassmannian parameters by Euclidean coordinates, thereby allowing the direct use of PyTorch optimizers [169] and avoiding direct Riemannian updates of these parameters.

A.3.3 Riemannian Multinomial Logistic Regression

A.3.3.1 RMLR on the SPD Manifold

This subsection offers additional details on the experiments on SPD MLRs.

Backbone networks and LogEig MLR. The SPDNet, TSMNet, SPDNetBN, SPDDSMNB, and SPDGCN backbones are reviewed in Sec. A.2.1, while RResNet is reviewed in Sec. A.2.5. In the SPD baseline models considered here, the Euclidean MLR in the codomain of matrix logarithm (matrix logarithm + FC + softmax) is used for classification. Following the terminology introduced in Sec. 4.2.4, we call this classifier the **LogEig MLR**. The LogEig MLR is the Euclidean classifier in the tangent space at the identity, which might distort the innate geometry of the SPD manifold.

SPD modeling and preprocessing. We use the same SPD modeling and preprocessing as the LieBN experiments in Sec. A.3.1.1.

Implementation details. For SPDNet [106] and TSMNet [123], we follow the official PyTorch code of SPDNetBN¹⁴ and TSMNet¹⁵ to implement our experiments. To evaluate the performance of our intrinsic classifiers, we substitute the LogEig MLR in SPDNet and TSMNet with our SPD MLRs. We implement our SPD MLRs induced by five parameterized metrics. On the Radar and HDM05 data sets, the learning rate is 10^{-2} , and the batch size is 30. On the Hinss2021 data set, following Kobler et al. [123], the learning rate is 10^{-3} with a 10^{-4} weight decay, and the batch size is 50. The maximum numbers of training epochs are 200, 200, and 50, respectively. We use the standard cross-entropy loss as the training objective and optimize the parameters with the Riemannian AMSGrad optimizer [18].

RResNet [117]. We focus on the AIM-based RResNet and use the official code¹⁶ and suggested network settings to implement the experiments with RResNet. We conduct 10-fold and 5-fold experiments on the HDM05 and NTU60 data sets, respectively. Since RResNet is developed based on SPDNet, we use the same learning settings as SPDNet for the action recognition task and borrow the best (θ, α, β) from Tab. 4.4 for our SPD MLRs under the RResNet backbone.

NTU60 SPD modeling. For NTU60 [178], each frame contains the 3D coordinates of 25 body joints and is therefore represented by a $25 \times 3 = 75$ -dimensional coordinate vector. Following Katsman et al. [117], each sequence is modeled as a 75×75 temporal covariance matrix, and evaluation follows the cross-view protocol [178].

¹⁴https://proceedings.neurips.cc/paper_files/paper/2019/file/6e69ebbfad976d4637bb4b39de261bf7-Supplemental.zip

¹⁵<https://github.com/rkobler/TSMNet>

¹⁶<https://github.com/CUAI/Riemannian-Residual-Neural-Networks>

Data Sets	(θ, α, β) -AIM	(θ, α, β) -EM	(α, β) -LEM	2θ -BWM	θ -LCM
Disease	(0.25,1,0)	(0.25,1,0)	(1,1)	0.25	0.5
Cora	(0.5,1,0)	(0.25,1,1/9)	(1,1/9)	0.25	0.5
Pubmed	(0.5,1,0)	(0.5,1,0)	(1, -1/3)	0.25	0.5

 Table A.5: (θ, α, β) of SPD MLRs on the SPDGCN backbone.

SPDGCN [231]. We use the official code¹⁷ and the suggested network settings in Zhao et al. [231]. Note that SPDGCN with SPD MLR retains the same network settings as vanilla SPDGCN. Tab. A.5 presents the hyperparameters (θ, α, β) on different data sets.

Network architectures. We denote the network architecture as $[d_0, d_1, \dots, d_L]$, where the dimension of the parameter in the i -th BiMap layer (Sec. A.2.1) is $d_i \times d_{i-1}$. For SPDNet, we also validate our SPD MLRs under different network architectures on the Radar and HDM05 data sets. The network architectures on the Radar data set are $[20, 16, 8]$ for the 2-block configuration and $[20, 16, 14, 12, 10, 8]$ for the 5-block configuration, while on the HDM05 data set, the network architectures are $[93, 30]$ for 1-block, $[93, 70, 30]$ for 2-block, and $[93, 70, 50, 30]$ for 3-block. For TSMNet, the 1-block architecture is $[40, 20]$.

Scoring metrics and evaluation protocols. We use the same scoring metrics and evaluation protocols as Sec. A.3.1.1. For the graph-learning experiments, following Zhao et al. [231], we report the 10-fold average and maximum node-classification accuracy.

Hyperparameters. We implement the SPD MLRs induced by not only five standard metrics, *i.e.*, LEM, AIM, EM, LCM, and BWM, but also five families of parameterized metrics. Therefore, in our SPD MLRs, we have a maximum of three hyperparameters, *i.e.*, θ, α, β , where (α, β) are associated with $O(n)$ -invariance and θ controls deformation. For (α, β) in (θ, α, β) -LEM, (θ, α, β) -AIM, and (θ, α, β) -EM, recalling Eq. (2.93), α is a scaling factor, while β measures the relative significance of traces. As scaling is less important [192], we set $\alpha = 1$. As for the value of β , we select it from a predefined set: $\{1, 1/n, 1/n^2, 0, -1/n + \epsilon, -1/n^2\}$, where n is the dimension of the input SPD matrices in SPD MLRs. The purpose of including $\epsilon \in \mathbb{R}_+$ is to ensure the positive definiteness of the inner product, *i.e.*, $\alpha + n\beta > 0$ and hence $(\alpha, \beta) \in \mathbf{ST}$. These chosen values for β allow for amplifying, neutralizing, or suppressing the trace components, depending on the characteristics of the data sets. For the deformation factor θ , we roughly select its value around its deformation boundary, *i.e.*, $[0.25, 1.5]$

¹⁷<https://github.com/andyweizhao/SPD4GNNs>

A.3. Experimental Details

Metric	(θ, α, β) -AIM	(θ, α, β) -EM	θ -LCM	2θ -BWM
Candidate Values	$\{0.25, 0.5, 0.75, 1, 1.25, 1.5\}$	$\{0.25, 0.5, 1, 1.5\}$	$\{0.5, 1, 1.5\}$	$\{0.25, 0.5, 0.75\}$

Table A.6: Candidate values for hyperparameters in SPD MLRs.

for (θ, α, β) -AIM, $[0.5, 1.5]$ for θ -LCM, $[0.25, 1.5]$ for (θ, α, β) -EM, and $[0.25, 0.75]$ for 2θ -BWM. The detailed values are listed in Tab. A.6.

A.3.3.2 RMLR on Rotation Matrices

LieNet backbone. The LieNet backbone is reviewed in Sec. A.2.3. In the official MATLAB implementation, the LogMap layer uses the Euler axis-angle representation. Classification is performed using the Euler axis-angle representation, followed by an FC layer and a softmax layer. As the axis-angle representation is equivalent to the matrix logarithm, we call this classifier **LogEig MLR** as well. This classifier is, therefore, also non-intrinsic.

Preprocessing. We use the shared rotation-matrix modeling, preprocessing, LieNet implementation, and SGD optimizer choice described in Sec. A.3.1.2. Following Huang et al. [107], the Lie MLR experiments use the G3D and HDM05 data sets. We trim HDM05 by removing under-represented sequences, resulting in 2,326 sequences across 122 classes.

Lie MLR. We use our Lie MLR to replace the axis-angle classifier in LieNet and call the resulting network LieNet+LieMLR. To alleviate the computational burden, we set each P_k to have shape $[\text{num}, 3, 3]$, where num is the spatial dimension of the input of the Lie MLR layer. In other words, P_k is shared in the temporal dimension. We adopt PyTorch3D [174] to calculate the matrix logarithm. Due to the instability of `pytorch3d.transforms.so3_log_map`, we first use `pytorch3d.transforms.matrix_to_axis_angle` to calculate the rotation axis and angle and then convert this representation into the matrix logarithm¹⁸.

Training details. Following Huang et al. [107], we focus on the 3-block and 2-block architectures for the G3D and HDM05 data sets, respectively, which are the suggested architectures for these two data sets. The learning rate is 10^{-2} on both data sets, and we further set the weight decay to 10^{-5} on the G3D data set. For LieNet and LieNet+LieMLR, we use `torch.nn.utils.clip_grad_norm_` for gradient clipping with a clipping factor of 5. The clipping is imposed on the dimensionality reduction weight in the final FC linear layer of LieNet or, accordingly, $A = \{A_1, \dots, A_C\}$ in the

¹⁸<https://github.com/facebookresearch/pytorch3d/issues/188>

Lie MLR layer of LieNet+LieMLR.

Scoring metrics. For the G3D data set, following LieNet [107], we adopt a 10-fold cross-subject test setting, where half the subjects are used for training and the other half are employed for testing. For the HDM05 data set, following Huang et al. [107], we randomly select half of the sequences for training and the rest for testing. Due to the instability of LieNet, we conduct 20-fold experiments and select the best 10 folds to evaluate the performance.

A.3.4 Proper Velocity Neural Networks

Common Implementations. We use the trivialization strategy reviewed in Sec. 2.7 in our MLR, FC, and GyroBN layers. Consequently, all trainable manifold-valued parameters in PVNN are represented by Euclidean coordinates and optimized using standard Euclidean optimizers.

A.3.4.1 Image Classification

The CIFAR-10 and CIFAR-100 data sets and their PyTorch splits are described in Sec. A.1.3. Following Bdeir et al. [15], we use data augmentation that includes random cropping with padding of 4 pixels and random horizontal flipping.

Implementation Details. We implement the experiments using the official code¹⁹ of Bdeir et al. [15]. All models share a common backbone, which consists of a ResNet-18 encoder followed by a hyperbolic MLR classifier. Except for PV MLR without Exp_0 , the output embedding of the ResNet-18 backbone is mapped to the target hyperbolic space via the exponential map at the identity e , that is, $\text{Exp}_e(x)$. Here, $e = \mathbf{0}$ for the Poincaré and PV spaces, and $e = \bar{\mathbf{0}}$ for the hyperboloid model. All models are trained from scratch. Optimization is performed using SGD [175] with an initial learning rate of 0.1, a momentum of 0.9, and a weight decay of 5×10^{-4} . Training is conducted with a batch size of 128 for 200 epochs. The learning rate is decayed by a factor of $\gamma = 0.2$ at epochs 60, 120, and 160. The curvature for the PV space is set as $K = -0.5$.

A.3.4.2 Graph Learning

The descriptions and summary statistics of the Disease, Airport, PubMed, and Cora data sets are collected in Sec. A.1.4.

Implementation Details. We adopt the official code of HGNC²⁰ [38] to conduct

¹⁹<https://github.com/kschwethelm/HyperbolicCV>

²⁰<https://github.com/HazyResearch/hgcn>

Hyperparameter	Disease	Airport	PubMed	Cora
Learning rate	0.01	0.01	0.05	0.05
Dropout	0.4	0.4	0.6	0.6
Curvature	-0.3	-0.3	-1.0	-1.0

Table A.7: Hyperparameters for PVNN that vary across graph data sets.

Setting	Epochs	Batch size	Weight decay
Value	2000	128	5×10^{-4}

Table A.8: Hyperparameters for PVNN that are shared across graph data sets.

experiments. The features of each node are embedded into the hyperbolic space via the exponential map at the identity. The hyperbolic network consists of two FC layers: the first maps the input feature dimension to a 16-dimensional hidden representation, and the second maps from 16 to 16. Each FC layer is followed by an activation function. An MLR layer is then used for classification. All models are trained using the Adam optimizer [122]. We evaluate performance every 10 epochs and employ early stopping with a patience of 200 evaluations, restoring the checkpoint with the best test accuracy. Tabs. A.7 and A.8 summarize the hyperparameters for PVNN. For KNN [147], HNN [76], HNN++ [181], and LNN [15], we follow their original papers to implement the experiments. Tab. A.9 summarizes the hyperbolic layers used in each model.

A.3.4.3 Genomic Sequence Learning

The description and complete summary of TEB are collected in Sec. A.1.5. We focus on five of its seven data sets.

Implementation Details. For the Euclidean CNN and the hyperbolic CNN baseline (HCNN-S), we directly use the results reported in the original paper [119, Tab. 2]. Our PVCNN architecture follows their implementation²¹. Each DNA sequence is represented as a length- L sequence with 4 input channels. We first apply a PV convolution that maps the 4 input channels to 32 channels, followed by PV TBN and a tanh tangent activation. A second PV convolution layer is then applied. The final PV feature is concatenated and passed through an FC layer, and finally classified with a PV MLR head. The curvature is initialized at $K = -0.5$ and learned during training. We train for 100 epochs with a step learning-rate schedule, using milestones at epochs 60 and

²¹<https://github.com/rrkhan/HGE>

Model	FC layer	Activation	MLR
PVNN	PV FC in Thm. 127	$\text{Exp}_0(\sigma(\text{Log}_0(x)))$	PV MLR in Thm. 126
KNN	$\text{Log}_0(W \text{Exp}_0(x))$	$\text{Exp}_0(\sigma(\text{Log}_0(x)))$	Euclidean MLR after Exp_0
HNN	$\text{Log}_0(W \text{Exp}_0(x))$	$\text{Exp}_0(\sigma(\text{Log}_0(x)))$	Poincaré MLR [76]
HNN++	Poincaré FC [181]	$\text{Exp}_0(\sigma(\text{Log}_0(x)))$	Poincaré MLR [181]
LNN	Lorentz FC [45]	Lorentz activation [15]	Lorentz MLR [15]

Table A.9: Summary of the hyperbolic layers used in the graph node classification models.

Setting	Value
Optimizer	Adam
Learning rate	$1e^{-4}$
Weight decay	$2e^{-2}$
Batch size	100
Dropout	0.1
Adam (β_1, β_2)	(0.9, 0.999)

Table A.10: Hyperparameters for TEB.

85 with a decay factor of 0.1. For the PV FC layer, σ in Eq. (5.29) is set to $\tanh$. All other hyperparameters are summarized in Tab. A.10.

A.3.5 Hyperbolic Busemann Neural Networks

A.3.5.1 Image Classification

Implementation Details. For CIFAR-10/100 and Tiny-ImageNet, we follow Bdeir et al. [15, App. C.1]. For ImageNet-1k, we follow Guo et al. [91, Sec. 4]. Tab. A.11 summarizes the data-set-specific hyperparameters. For the hyperbolic MLR, before mapping into the hyperbolic space, we clip the feature vector by

$$\text{CLIP}(x; r) = \min \left\{ 1, \frac{r}{\|x\|} \right\} x \quad (\text{A.27})$$

where $r > 0$ is a hyperparameter. The clipped Euclidean embedding is projected via the exponential map to the target hyperbolic space: $\text{Exp}_e(\text{CLIP}(x; r))$. For the Lorentz model, the clipping parameter is $r = 1$ on CIFAR-10/100 and $r = 4$ on Tiny-ImageNet and ImageNet-1k. On the Poincaré ball, $r = 1$ on all four data sets.

All methods are implemented in PyTorch and trained with cross-entropy loss. The results of MLR, PMLR, and LMLR on CIFAR-10/100 and Tiny-ImageNet are copied

Hyperparameter	CIFAR-10/100 Tiny-ImageNet	ImageNet-1k
Epochs	200	100
Batch size	128	256
Initial learning rate	0.1	0.1
LR schedule	60, 120, 160; $\gamma = 0.2$	30, 60, 90; $\gamma = 0.1$
Weight decay	$5e^{-4}$	$1e^{-4}$
Optimizer	SGD	SGD
Precision	32-bit	32-bit
Curvature K	-1	-1

Table A.11: Summary of hyperparameters used in the image classification task.

from Bdeir et al. [15, Tab. 1], while those of PBMLR-P on CIFAR-10/100 are copied from Nguyen et al. [162, Tab. 2]. The remaining results are obtained by our careful implementation.

A.3.5.2 Genome Sequence Learning

Implementation Details. We mainly follow the official implementations of Khan et al. [119] for data processing, model architecture, and training. We adopt a simple CNN with three convolutional blocks followed by dense, ReLU-activated layers to extract features [119, Fig. 4]. Before classification, the features are clipped and mapped to the target hyperbolic space as in Eq. (A.27), then passed to either a prior hyperbolic MLR or our BMLR head. The clipping factor defaults to $r = 1$, with the following exceptions on GUE: on the Lorentz model, $r = 2.0$ for Covid variant classification and $r = 5.0$ for species classification; on the Poincaré ball, $r = 2.0$ for species classification. Following Khan et al. [119], we treat the curvature K as a learnable parameter initialized as -1 . The remaining hyperparameters are listed in Tab. A.12. All models share these hyperparameters, except LMLR on Covid variant classification, for which we set the weight decay to $1e^{-3}$ to ensure convergence.

All methods are implemented in PyTorch and trained with cross-entropy loss. Results are obtained from our reimplementation.

A.3.5.3 Node Classification

Implementation Details. We follow the official implementations of HGCN [38] and PBMLR [162] and conduct experiments on both the Poincaré ball and the Lorentz model. We adhere to their experimental settings. The only changes are weight decay

Batch size	100
Epochs	100
Optimizer	Adam
β_1, β_2	0.9, 0.999
Initial learning rate	$1e^{-4}$
LR schedule	60, 85; $\gamma = 0.1$
Weight decay	0.1
Initial curvature K	-1

Table A.12: Hyperparameters for genome sequence learning.

Space	Weight decay	Dropout
$\mathbb{P}_K^n$	$1e-4, 1e-5, 1e-3, 1e-3$	0.3, 0, 0, 0.2
$\mathbb{L}_K^n$	$1e-4, 5e-5, 1e-3, 1e-3$	0, 0, 0, 0.3

Table A.13: Hyperparameters for node classification on Disease, Airport, PubMed, and Cora.

and dropout. We train with cross-entropy loss and the Adam optimizer [122] for 5000 epochs with a learning rate of $1e^{-2}$, curvature set to -1 , embedding dimension 16, and three GCN layers. We tune weight decay and dropout and report the values in Tab. A.13. All methods are implemented in PyTorch. The results of HGCN-PMLR and HGCN-PBMLR-P on the Poincaré ball are taken from Nguyen et al. [162, Tab. 11]. Results for the remaining baselines are obtained from our reimplementation following the original settings.

A.3.5.4 Link Prediction

Implementation Details. We follow the official implementations of HNN [76], HNN++ [181], and HyboNet [45], and adopt the experimental protocol of Chami et al. [38] for link prediction. The encoder consists of two fully connected layers: the first maps the input features to 16, and the second maps 16 to 16. Each FC layer is instantiated as either our BFC or an existing hyperbolic FC layer. After each FC, we apply the activation $\text{Exp}_e(\text{ReLU}(\text{Log}_e(x)))$, where e denotes the model origin. Following Chami et al. [38], this activation is disabled on Cora. As in the Möbius layer, we apply a gyro bias after each FC, that is, $x \oplus_{\mathcal{H}} b$. We train with Adam [122] at a learning rate of $1e^{-2}$ and tune weight decay and FC dropout. For BFC, we use $\phi = \tanh$ on Airport and Cora and the identity map on the other two data sets.

A.3.6 Full-Rank Correlation Networks

The Radar, HDM05, FPFA, and NTU120 data sets are described in Sec. A.1. For HDM05 and FPFA, each sequence is normalized for body-part length, scale, and view using the preprocessing of Vemulapalli et al. [203]. For NTU120, we follow Chen et al. [46] to preprocess the data.

A.3.6.1 Input Data

Correlation Input in CorNets. Following Wang et al. [210], Nguyen et al. [161], we first model each sample as a multi-channel SPD tensor. For HDM05 and FPFA, the skeleton preprocessing and multi-channel covariance construction follow the GyroGr pipeline of Nguyen and Yang [159]; CorNet subsequently converts every covariance matrix into a full-rank correlation matrix by

$$\text{Cor} : \mathcal{S}_{++}^n \ni \Sigma \mapsto C = \mathbb{D}(\Sigma)^{-\frac{1}{2}} \Sigma \mathbb{D}(\Sigma)^{-\frac{1}{2}} \in \text{Cor}^+(n). \quad (\text{A.28})$$

For Radar, we follow Wang et al. [210] and use temporal convolution followed by covariance pooling to obtain a multi-channel covariance tensor of shape $[c, 20, 20]$. After preprocessing, the input correlation tensor shapes are $[7, 20, 20]$, $[3, 28, 28]$, $[9, 28, 28]$, and $[6, 28, 28]$ on Radar, HDM05, FPFA, and NTU120, respectively.

A.3.6.2 Implementation Details

SPD Baselines. We follow the official PyTorch code of SPDNetBN²² to implement SPDNet and SPDNetBN. For LieBN²³, we focus on the instantiation under LCM [137], while for RResNet²⁴, we implement the ones induced by AIM [171] and LEM [9]. For SPD MLR²⁵, we implement the one based on LCM. Due to the lack of official code, Gyro-based models are carefully reimplemented from their original papers. Following Nguyen et al. [161], GyroSPD++ combines an AIM-based convolution with an LEM-based MLR.

Grassmannian Baselines. Since GrNet is officially implemented in MATLAB, we carefully re-implemented it using PyTorch. Additionally, as both GyroGr and GyroGr-Scaling do not release official code, we re-implemented them based on the original paper

²²https://proceedings.neurips.cc/paper_files/paper/2019/file/6e69ebbfad976d4637bb4b39de261bf7-Supplemental.zip

²³<https://github.com/GitZH-Chen/LieBN>

²⁴<https://github.com/CUAI/Riemannian-Residual-Neural-Networks>

²⁵<https://github.com/GitZH-Chen/SPDMLR>

Data Set	Model	Optimizer	lr	wd	Matrix Power	Converged Epoch
Radar	CorNet-ECM	Adam	$1e^{-2}$	N/A	1.5	50
	CorNet-LECM	Adam	$1e^{-2}$	N/A	-0.25	50
	CorNet-OLM	Adam	$1e^{-2}$	N/A	-0.25	50
	CorNet-LSM	Adam	$1e^{-2}$	N/A	0.75	50
	CorNet-PHCM	Adam	$1e^{-2}$	N/A	0.75	50
HDM05	CorNet-ECM	Adam	$1e^{-3}$	$1e^{-3}$	0.125	100
	CorNet-LECM	Adam	$1e^{-4}$	$1e^{-3}$	0.5	150
	CorNet-OLM	SGD	$5e^{-2}$	$1e^{-3}$	0.25	200
	CorNet-LSM	Adam	$1e^{-3}$	N/A	-0.75	50
	CorNet-PHCM	Adam	$1e^{-2}$	N/A	-0.25	50
FPHA	CorNet-ECM	Adam	$5e^{-3}$	N/A	-0.25	150
	CorNet-LECM	Adam	$5e^{-4}$	$1e^{-4}$	-0.5	150
	CorNet-OLM	Adam	$1e^{-4}$	N/A	-1	50
	CorNet-LSM	Adam	$1e^{-3}$	N/A	-1	50
	CorNet-PHCM	Adam	$1e^{-3}$	$1e^{-4}$	-0.5	150
NTU120	CorNet-ECM	SGD	$1e^{-2}$	N/A	0.25	50
	CorNet-LECM	SGD	$1e^{-2}$	N/A	0.25	50
	CorNet-OLM	SGD	$5e^{-3}$	N/A	0.25	50
	CorNet-LSM	SGD	$1e^{-3}$	N/A	0.25	50
	CorNet-PHCM	Adam	$1e^{-3}$	N/A	0.25	50

Table A.14: Hyperparameters in CorNets.

[159]. For all Grassmannian comparative methods, we use SGD [175] with a learning rate of $5e^{-2}$.

CorNets. On all four data sets, we employ a single convolutional kernel for global convolution, *i.e.*, applying a global receptive field across the channel dimension. The output dimensions of the correlation convolutional layer are 8×8 , 26×26 , 26×26 , and 11×11 for the Radar, HDM05, FPHA, and NTU120 data sets, respectively.

We primarily use the Adam [122] and SGD [175] optimizers. Inspired by the deformation effect on the latent SPD geometries by the matrix power over the SPD manifold illustrated in Fig. 4.2, we apply the matrix power before correlation modeling ($\text{Cor}(\cdot)$) as activation. In particular, when the data are centered at zero and power is -1 , $\text{Cor}(\Sigma^{-1})$ corresponds to the partial correlation matrix of the covariance matrix Σ [191, Lem. 1.6]. The batch size is set to 30, and training is capped at 200 epochs, although most cases converge in fewer than 150 epochs. Due to the different correlation geometries, the hyperparameters vary for CorNets under different geometries. Tab. A.14 summarizes all the hyperparameters.

Extra Computational Details for OLM and LSM Layers. For the MLR, FC, and convolutional layers induced by OLM and LSM, the key computations involve Exp° and Log^* , which depend on the calculations of $\mathcal{D}$ and $\mathcal{D}^*$. In our experiments, we empirically observe that iterating until convergence is more effective for $\mathcal{D}$, whereas a single step of Newton’s method generally performs best for $\mathcal{D}^*$. Accordingly, we set $\mathcal{D}$ to iterate until convergence, leveraging Thm. 145 for accurate backpropagation. For

$\mathcal{D}^*$, we adopt a single iteration in Newton’s method and use automatic differentiation (autograd) through this single step for backpropagation.

A.3.6.3 Additional Details on Visualization

We provide additional interpretations of the low-dimensional visualization and clarify how the decision hyperplanes in Fig. 5.5 are obtained. The correlation-in-SPD visualization is discussed in Sec. 2.9.2.

Visualization of Low-Dimensional SPD and Correlation Matrices. Any 2×2 covariance matrix in $\mathcal{S}_{++}^2$ can be written as

$$\Sigma = \begin{pmatrix} a & b \\ b & d \end{pmatrix}, \quad a > 0, \ d > 0, \ ad - b^2 > 0. \quad (\text{A.29})$$

Embedding Σ into $\mathbb{R}^3$ via the map $\Sigma \mapsto (a, b, d)$ identifies $\mathcal{S}_{++}^2$ with the interior of the quadratic cone

$$\{(a, b, d) \in \mathbb{R}^3 \mid a > 0, \ d > 0, \ ad - b^2 > 0\}, \quad (\text{A.30})$$

which is an open cone in $\mathbb{R}^3$.

For 2×2 correlation matrices, any $C \in \text{Cor}^+(2)$ has the form

$$C = \begin{pmatrix} 1 & r \\ r & 1 \end{pmatrix}, \quad r \in (-1, 1). \quad (\text{A.31})$$

Thus, $\text{Cor}^+(2)$ is one-dimensional. Embedding C into $\mathbb{R}^3$ as $(1, r, 1)$ yields a line segment inside the cone corresponding to $\mathcal{S}_{++}^2$.

For 3×3 correlation matrices, any $C \in \text{Cor}^+(3)$ is parameterized by its off-diagonal entries (r_{12}, r_{13}, r_{23}) :

$$C = \begin{pmatrix} 1 & r_{12} & r_{13} \\ r_{12} & 1 & r_{23} \\ r_{13} & r_{23} & 1 \end{pmatrix}. \quad (\text{A.32})$$

Embedding C into $\mathbb{R}^3$ via $C \mapsto (r_{12}, r_{13}, r_{23})$ produces an open ellipsope in $\mathbb{R}^3$. This is the representation of $\text{Cor}^+(3)$ used in Fig. 5.5, where each point in the ellipsope corresponds to one 3×3 correlation matrix.

Construction of Fig. 5.5. For ECM, LECM, OLM, and LSM, the decision hyperplane in the correlation MLR is the Riemannian hyperplane in Sec. 4.3.2.1 specialized

to $\mathcal{M} = \text{Cor}^+(n)$:

$$H_{A,P} = \{X \in \text{Cor}^+(n) \mid \langle \text{Log}_P(X), A \rangle_P = 0\}, \quad P \in \text{Cor}^+(n), A \in T_P \text{Cor}^+(n). \quad (\text{A.33})$$

In Fig. 5.5, we focus on $\text{Cor}^+(3)$ and visualize it as the open ellipsope in $\mathbb{R}^3$ via the embedding

$$C \in \text{Cor}^+(3) \mapsto (C_{21}, C_{31}, C_{32}) \in \mathbb{R}^3. \quad (\text{A.34})$$

Given a Log-Euclidean metric and parameters (A, P) , each correlation matrix C is first mapped to the tangent space at P by $\text{Log}_P(C)$, and we evaluate the linear form $\langle \text{Log}_P(C), A \rangle_P$. The set of points in the ellipsope where this scalar equals zero corresponds to the decision hyperplane $H_{A,P}$ and is plotted as the separating surface.

For PHCM, the margin hyperplane is defined in the β -concatenated Poincaré embedding. Let Φ be the diffeomorphism in Eq. (5.79) that maps $C \in \text{Cor}^+(n)$ to the poly-Poincaré space $\mathbb{P}^{n-1}$, and let β -concatenation be the Poincaré operation in Sec. 5.4.3.2. We define

$$\tilde{x}(X) = \beta\text{-concat}(\Phi(X)) \in \mathbb{P}^N, \quad N = \frac{n(n-1)}{2}, \quad (\text{A.35})$$

and the PHCM hyperplane

$$H_{a,p} = \{X \in \text{Cor}^+(n) \mid \langle \text{Log}_p(\tilde{x}(X)), a \rangle_p = 0\}, \quad p \in \mathbb{P}^N, a \in T_p \mathbb{P}^N. \quad (\text{A.36})$$

Here $\mathbb{P}^N = \{x \in \mathbb{R}^N \mid \|x\|^2 < 1\}$ is the N -dimensional unit Poincaré ball. In Fig. 5.5, we first map each correlation matrix $C \in \text{Cor}^+(3)$ to $\tilde{x}(C) \in \mathbb{P}^N$, apply the Poincaré logarithm Log_p at a reference point p , and then visualize the zero level set of the linear form $\langle \text{Log}_p(\tilde{x}(C)), a \rangle_p$ as the PHCM decision hyperplane.

A.3.7 Adaptive Log-Euclidean Metrics

A.3.7.1 Data Sets and Settings

Although the proposed ALog layers can be plugged into existing SPD networks, we focus on the SPDNet framework [106]. We follow the PyTorch code provided by SPDNetBN²⁶ to reproduce SPDNet and SPDNetBN and implement our approaches.

Following previous work [106, 31], we evaluate our methods on HDM05 [153], FPHA [80], and AFEW [68]. The data set descriptions are given in Sec. A.1. We use the HDM05 and FPHA temporal covariance representations and the filtered HDM05 setting

²⁶SPDNetBN supplementary code

specified in Sec. A.3.1.1. For AFEW, we use the released pre-trained FAN²⁷ [149] to extract deep features and establish a 512×512 temporal covariance matrix for each video.

We denote the dimensions of the transformation layers in the SPDNet backbone by $\{d_0, d_1, \dots, d_L\}$. Following the settings in Brooks et al. [31], all networks are trained using the default RSGD [17] with a fixed learning rate γ and a batch size of 30. To make ALog start from the vanilla matrix logarithm, the parameters in MUL, DIV, and RELU are initialized as 1, 1, and e , respectively. By abuse of notation, SPDNet-ALog-MUL is abbreviated as ALog-MUL, denoting that we substitute the LogEig layer in SPDNet with the proposed ALog optimized by MUL.

A.3.7.2 Implementation Details of Additional Applications

NTU60 data set. For NTU60, we use the cross-view protocol and 75×75 covariance representation specified in Sec. A.3.3.1.

As reported in Sec. 6.2.4, MUL shows the best performance. Therefore, we view A in Eq. (6.31) as the parameter for all experiments. In the following, we discuss in detail the specific implementation of each method.

LieBN. We follow the official code²⁸ to implement the experiments. The learning rate is $5e^{-2}$. Since our LieBN-ALEM shows early convergence, we set the numbers of training epochs to 150, 50, and 30 for the [93, 30], [93, 70, 30], and [93, 70, 50, 30] architectures. Other settings are the same as Sec. 6.2.4.

RResNet. We follow the official code²⁹ to implement the experiments. For the HDM05 data set, we use RSGD [17] with a $5e^{-2}$ learning rate for 200 training epochs. For the NTU60 data set, we use Riemannian AMSGrad [17] with a $1e^{-2}$ learning rate for 50 training epochs. We adopt the architectures of [93, 30] and [75, 30] on these two data sets.

Gyro MLR. Since the code of gyro MLR is not publicly available, we carefully re-implement the gyro MLR in Nguyen and Yang [159]. We adopt an architecture of [75, 30] under an SGD optimizer. The batch size and number of training epochs are 30 and 200, respectively.

²⁷<https://github.com/Open-Debin/Emotion-FAN>

²⁸<https://github.com/GitZH-Chen/LieBN>

²⁹<https://github.com/CUAI/Riemannian-Residual-Neural-Networks>

A.3.8 Product Cholesky Metrics

A.3.8.1 Implementation Details of SPD Neural Networks

Backbone networks. SPDNet and GyroSPD are reviewed in Sec. A.2.1, while RResNet is reviewed in Sec. A.2.5.

Data sets and preprocessing. The Radar, HDM05, and FPHA data sets are described in Sec. A.1. The global covariance representations for SPDNet and RResNet follow Sec. A.3.1.1; the GyroSPD-specific multichannel construction is given below.

SPD MLR. We follow the official PyTorch code³⁰ to implement the SPD MLR developed in Sec. 4.2.2. Due to the lack of official code, the GyroSPD backbone is carefully reimplemented in PyTorch following the original paper [159]. For simplicity, we set $\mathbb{M}$ in $(\theta, \mathbb{M})$ -BWCM to the identity matrix.

SPDNet. Following Huang and Van Gool [106], we replace the vanilla tangent classifier (LogEig + FC + softmax) in SPDNet with SPD MLRs induced by AIM, LEM, LCM, θ -PCM, and θ -BWCM. We use a Riemannian AMSGrad [17] with a learning rate of $1e^{-2}$, a batch size of 30, and a maximum of 200 epochs. We denote the architectures by $[d_0, d_1, \dots, d_L]$, where d_i is the output dimension of the i -th BiMap layer. Following prior work [106, 31], we adopt [20, 16, 8] on Radar, [63, 33] on FPHA, and [93, 30], [93, 70, 30], [93, 70, 50, 30] for 1-, 2-, and 3-block variants on HDM05. As shown in Tab. 3.9c, matrix power improves the performance of Cholesky-based metrics on FPHA. We therefore apply a power of -0.25 before SPD MLR layers under LCM, θ -PCM, and θ -BWCM. For θ -PCM on FPHA, we further adopt a weight decay of $1e^{-4}$. The deformation factor θ is reported in Tab. A.15.

GyroSPD. Following Nguyen and Yang [159], the backbone consists of one gyro-translation layer followed by an SPD MLR. We compare metrics based on LEM, LCM, AIM, and our proposed geometries under the same settings: Riemannian AMSGrad with a learning rate of $1e^{-2}$ and a batch size of 30. The models are trained for up to 100, 100, and 50 epochs on Radar, HDM05, and FPHA, respectively. Similarly, a matrix power of 0.25 is applied on FPHA for LCM and our Cholesky-based metrics. The deformation factor θ is reported in Tab. A.15.

SPD input of SPDNet. We use the global covariance representations specified in Sec. A.3.1.1.

SPD input of GyroSPD. For Radar, the input is the same as SPDNet. For HDM05 and FPHA, we follow Nguyen and Yang [159] to model each sample into a multi-channel covariance tensor $[c, n, n]$. Specifically, we first identify the closest left

³⁰<https://github.com/GitZH-Chen/SPDMLR>

Backbone	Metric	Radar	HDM05	FPHA
SPDNet	θ -PCM	-1.5	-0.5	0.75
	θ -BWCM	-0.75	-1.5	1
GyroSPD	θ -PCM	-0.75	-0.75	-0.75
	θ -BWCM	-1.5	-1.5	-0.5

Table A.15: Hyperparameter θ in θ -PCM and θ -BWCM. It is selected from the candidate values used for SPDNet and GyroSPD.

(right) neighbor of each joint based on its distance to the hip (wrist) joint, and then combine the 3D coordinates of each joint and those of its left (right) neighbor to create a feature vector for the joint. For a given frame t , we compute its Gaussian embedding [143]:

$$Y_t = (\det \Sigma_t)^{-\frac{1}{n+1}} \begin{bmatrix} \Sigma_t + \mu_t (\mu_t)^\top & \mu_t \\ (\mu_t)^\top & 1 \end{bmatrix}, \quad (\text{A.37})$$

where μ_t and Σ_t are the mean vector and covariance matrix computed from the set of feature vectors within the frame. The lower part of the matrix $\log(Y_t)$ is flattened to obtain a vector $\tilde{v}_t$. All vectors $\tilde{v}_t$ within a time window $[t, t + c - 1]$, where c is determined from a temporal pyramid representation of the sequence (the number of temporal pyramids is set to 2 in our experiments), are used to compute a covariance matrix as

$$\tilde{\Sigma}_t = \frac{1}{c} \sum_{i=t}^{t+c-1} (\tilde{v}_i - \bar{v}_t)(\tilde{v}_i - \bar{v}_t)^\top, \quad (\text{A.38})$$

where $\bar{v}_t = \frac{1}{c} \sum_{i=t}^{t+c-1} \tilde{v}_i$. The resulting $\{\tilde{\Sigma}_t\}$ are the covariance matrices that we need. On FPHA, we generate the covariances based on three sets of neighbors: left, right, and vertical (bottom) neighbors. After preprocessing, the input covariance matrices are $[3, 28, 28]$ and $[9, 28, 28]$ on HDM05 and FPHA, respectively.

RResNet implementation. The backbone architectures of RResNet [118] are similar to those of SPDNet: $[20, 16, 8]$ on Radar, $[93, 30]$ on HDM05, and $[63, 33]$ on FPHA, respectively. The only architectural difference lies in the head: SPDNet directly uses a classification layer, whereas RResNet attaches a residual block before the classification head. Following RResNet, we adopt $\text{Log}_{I_n} + \text{FC} + \text{softmax}$ for classification under each metric. We use the official code³¹ to re-implement the AIM- and LEM-based RResNet. For the RResNets based on LCM and our θ -PCM, we adopt the following settings: a learning rate of $1e^{-2}$, a batch size of 30, and a maximum of 200 epochs.

³¹<https://github.com/CUAI/Riemannian-Residual-Neural-Networks>

We use the standard cross-entropy loss as the training objective and optimize the parameters with the Riemannian AMSGrad optimizer [17]. The Cholesky diagonal power is set to -1 , 0.5 , and -0.5 on the Radar, HDM05, and FPHA data sets, respectively. Additionally, on the FPHA data set, a matrix power of -0.25 is applied before the residual blocks to activate the latent geometry for both LCM and our metric.

SPD input. The input SPD matrices are the same as those used in SPDNet.

A.4 Additional Discussions

A.4.1 Riemannian Multinomial Logistic Regression

A.4.1.1 RMLR as a Natural Extension of the Euclidean MLR

Proposition 180. *When $\mathcal{M} = \mathbb{R}^n$ is the standard Euclidean space, the RMLR defined in Thm. 114 becomes the Euclidean MLR in Eq. (4.1).*

Proof. On the standard Euclidean space $\mathbb{R}^n$, $\text{Log}_y x = x - y, \forall x, y \in \mathbb{R}^n$. Besides, the differential maps of left translation and parallel transport are the identity maps. Therefore, given $x, p_k \in \mathbb{R}^n$ and $a_k \in \mathbb{R}^n \setminus \{0\} \cong T_0 \mathbb{R}^n \setminus \{0\}$, we have

$$p(y = k \mid x \in \mathbb{R}^n) \propto \exp(\langle \text{Log}_{p_k} x, a_k \rangle_{p_k}), \quad (\text{A.39})$$

$$\propto \exp(\langle x - p_k, a_k \rangle), \quad (\text{A.40})$$

$$\propto \exp(\langle x, a_k \rangle - b_k), \quad (\text{A.41})$$

where $b_k = \langle p_k, a_k \rangle$. □

A.4.1.2 Gyro SPSP MLR as a Special Case of Our RMLR

Gyro SPSP MLR [161] is derived from the product of the Grassmannian and SPD gyro spaces. This section will show that the gyro SPSP MLR is a special case of our RMLR on the product geometry of the SPSP manifold. We first review some necessary results about gyro SPSP MLR and then show the equivalence.

Following the notations in Nguyen et al. [161], we denote the Grassmannian with canonical metric under the projector and ONB perspective as $\widetilde{\text{Gr}}(p, n)$ and $\text{Gr}(p, n)$, respectively. The space of $n \times n$ SPSP matrices with a fixed rank p , denoted as $\mathcal{S}_{n,p}^+$, forms an SPSP manifold [26]. As shown in Bonnabel et al. [26], Nguyen et al. [161], the SPSP manifold is a product space, *i.e.*, $\mathcal{S}_{n,p}^+ \cong \text{Gr}(p, n) \times \mathcal{S}_{++}^p$. In other words, every

$P \in \mathcal{S}_{n,p}^+$ can be decomposed as $P = U_P S_P U_P^\top$ with $U_P \in \text{Gr}(p, n)$ and $S_P \in \mathcal{S}_{++}^p$. We further denote $\mathcal{S}_{++}^{p,g}$ as the SPD manifold with metric g , where g could be AIM, LEM, or LCM. As shown in Nguyen et al. [161], the gyro space in $\mathcal{S}_{n,p}^+$ can be defined by the product of gyro spaces of $\text{Gr}(p, n)$ and $\mathcal{S}_{++}^{p,g}$. By this product structure, Nguyen et al. [161] proposed the SPSP Pseudo-gyrodistance to a hyperplane.

Definition 181 (SPSP hypergyroplanes [161]). Let $P, W \in \text{Gr}(p, n) \times \mathcal{S}_{++}^{p,g}$. Then hypergyroplanes in the structure space $\text{Gr}(p, n) \times \mathcal{S}_{++}^{p,g}$ are defined as

$$H_{W,P}^{psd,g} = \left\{ Q \in \text{Gr}(p, n) \times \mathcal{S}_{++}^{p,g} \mid \langle \ominus_{psd,g} P \oplus_{psd,g} Q, W \rangle^{psd,g} = 0 \right\}, \quad (\text{A.42})$$

where $\oplus_{psd,g}$ and $\langle \cdot, \cdot \rangle^{psd,g}$ are the gyro addition and gyro inner product, respectively, as defined in Nguyen et al. [161].

Theorem 182 (SPSP pseudo-gyrodistance [161]). Let

$$W = (U_W, S_W), \quad P = (U_P, S_P), \quad X = (U_X, S_X) \in \text{Gr}(p, n) \times \mathcal{S}_{++}^{p,g}, \quad (\text{A.43})$$

and let $H_{W,P}^{psd,g}$ be a hypergyroplane in the structure space $\text{Gr}(p, n) \times \mathcal{S}_{++}^{p,g}$. Then the pseudo-gyrodistance from X to $H_{W,P}^{psd,g}$ is given by

$$\bar{d}(X, H_{W,P}^{psd,g}) = \frac{\left| \lambda \left\langle (\tilde{\ominus}_{gr} U_P \tilde{\oplus}_{gr} U_X) (\tilde{\ominus}_{gr} U_P \tilde{\oplus}_{gr} U_X)^\top, U_W U_W^\top \right\rangle^{gr} + \langle \ominus_g S_P \oplus_g S_X, S_W \rangle^g \right|}{\sqrt{\lambda (\|U_W U_W^\top\|^{gr})^2 + (\|S_W\|^g)^2}}, \quad (\text{A.44})$$

where $\|\cdot\|^{gr}$ and $\|\cdot\|^g$ are the gyro norms on the Grassmann and SPD manifolds [161], and $\langle \cdot, \cdot \rangle^{gr}$ and $\langle \cdot, \cdot \rangle^g$ are gyro inner products [161]. $\tilde{\ominus}_{gr}$ and $\oplus_g$ are gyro additions on $\text{Gr}(p, n)$ and $\mathcal{S}_{++}^{p,g}$.

Denoting g^{gr} as the canonical metric on $\text{Gr}(p, n)$ and g as AIM, LEM, or LCM, we can prove that Thm. 182 is a special case of Thm. 113.

Theorem 183. Under the product metric $g^{psd,g} = \lambda g^{gr} \times g$, the Riemannian hyperplane in Eq. (4.19) on the SPSP manifold equals the SPSP hypergyroplane in Thm. 181. Similarly, the Riemannian margin distance in Thm. 113 on the SPSP manifold equals the SPSP Pseudo-gyrodistance in Thm. 182.

Proof. Following the notations in Thms. 181 and 182, we further denote $P = U_P S_P U_P^\top$,

$Q = U_Q S_Q U_Q^\top$, $W = U_W S_W U_W^\top$, and $X = U_X S_X U_X^\top$ with $U_P, U_Q, U_W, U_X \in \text{Gr}(p, n)$ and $S_P, S_Q, S_W, S_X \in \mathcal{S}_{++}^{p,g}$. I_p is the $p \times p$ identity matrix. $I_{p,n} = (I_p, 0)^\top$ is the gyro identity on $\text{Gr}(p, n)$. $\widetilde{\Gamma}^{gr}$, $\widetilde{\text{Log}}^{gr}$, and $\langle \cdot, \cdot \rangle_{U_P}^{gr}$ denote Riemannian parallel transport along a geodesic, the Riemannian logarithm, and the Riemannian metric at U_P on $\text{Gr}(p, n)$, respectively. Γ^g , Log^g , and $\langle \cdot, \cdot \rangle_{S_P}^g$ denote Riemannian parallel transport along a geodesic, the Riemannian logarithm, and the Riemannian metric at S_P on $\mathcal{S}_{++}^{p,g}$, respectively. $\Gamma^{psd,g}$, $\text{Log}^{psd,g}$, and $\langle \cdot, \cdot \rangle_X^{psd,g}$ denote Riemannian parallel transport along a geodesic, the Riemannian logarithm, and the Riemannian metric at X on $\mathcal{S}_{n,p}^+ \cong \text{Gr}(p, n) \times \mathcal{S}_{++}^p$, respectively.

First, we show that the SPSD hypergyroplane equals our Riemannian hyperplane in Eq. (4.19). We have the following:

$$\begin{aligned}
 & \langle \ominus_{psd,g} P \oplus_{psd,g} Q, W \rangle^{psd,g} \\
 & \stackrel{(1)}{=} \lambda \langle \ominus_{gr} U_P \oplus_{gr} U_Q, U_W \rangle^{gr} + \langle \ominus_g S_P \oplus_g S_Q, S_W \rangle^g \\
 & \stackrel{(2)}{=} \lambda \langle \text{Log}_{U_P}^{gr} U_Q, \tilde{A}_{U_W} \rangle_{U_P}^{gr} + \langle \text{Log}_{S_P}^g S_Q, \tilde{A}_{S_W} \rangle_{S_P}^g \\
 & \stackrel{(3)}{=} \langle \text{Log}_P^{psd,g} Q, \tilde{A} \rangle_P^{psd,g}
 \end{aligned} \tag{A.45}$$

where

$$\tilde{A}_{U_W} = \widetilde{\Gamma}_{I_{p,n} \rightarrow U_P}^{gr} \left(\widetilde{\text{Log}}_{I_{p,n}}^{gr} (U_W) \right), \tag{A.46}$$

$$\tilde{A}_{S_W} = \Gamma_{I_p \rightarrow S_P}^g \left(\text{Log}_{I_p}^g (S_W) \right), \tag{A.47}$$

$$\tilde{A} = (\tilde{A}_{U_W}, \tilde{A}_{S_W}) \in T_P \mathcal{S}_{n,p}^+ \cong T_{U_P} \text{Gr}(p, n) \times T_{S_P} \mathcal{S}_{++}^{p,g}, \tag{A.48}$$

where $\times$ denotes the Cartesian product. The above derivation comes from the following.

- (1) The definition of gyro addition, gyro inverse, and gyro inner product on the SPSD manifold [161, Sec. 3.3].
- (2) The proof by Nguyen et al. [161, Prop. 3.2] indicates that similar results also hold on the Grassmannian. Combining that proposition with its Grassmannian counterparts yields the equation.
- (3) The Riemannian product geometry.

By the product geometry of the SPSD manifold, we can immediately get

$$\tilde{A} = (\tilde{A}_{U_W}, \tilde{A}_{S_W}) = \Gamma_{\tilde{I}_{p,n} \rightarrow P}^{psd,g} \left(\text{Log}_{\tilde{I}_{p,n}}^{psd,g} (W) \right), \tag{A.49}$$

where $\tilde{I}_{p,n} = I_{p,n} I_{p,n}^\top$ is the gyro identity on the SPSP manifold.

Next, we show the equivalence between SPSP pseudo-gyrodistance and our Riemannian margin distance:

$$\begin{aligned}
\bar{d}(X, H_{W,P}^{psd,g}) &\stackrel{(1)}{=} \frac{|\langle \ominus_{psd,g} P \oplus_{psd,g} X, W \rangle^{psd,g}|}{\|W\|^{psd,g}} \\
&\stackrel{(2)}{=} \frac{|\langle \text{Log}_P^{psd,g} X, \tilde{A} \rangle_P^{psd,g}|}{\|W\|^{psd,g}} \\
&\stackrel{(3)}{=} \frac{|\langle \text{Log}_P^{psd,g} X, \tilde{A} \rangle_P^{psd,g}|}{\left\| \text{Log}_{\tilde{I}_{p,n}}^{psd,g}(W) \right\|_{\tilde{I}_{p,n}}^{psd,g}} \\
&\stackrel{(4)}{=} \frac{|\langle \text{Log}_P^{psd,g} X, \tilde{A} \rangle_P^{psd,g}|}{\left\| \Gamma_{\tilde{I}_{p,n} \rightarrow P}^{psd,g} \left(\text{Log}_{\tilde{I}_{p,n}}^{psd,g}(W) \right) \right\|_P^{psd,g}} \\
&\stackrel{(5)}{=} \frac{|\langle \text{Log}_P^{psd,g} X, \tilde{A} \rangle_P^{psd,g}|}{\left\| \tilde{A} \right\|_P^{psd,g}} \\
&\stackrel{(6)}{=} d(X, \tilde{H}_{\tilde{A},P})
\end{aligned} \tag{A.50}$$

- (1) The definition of gyro addition, gyro inverse, gyro inner product, and gyro norm on the SPSP manifold.
- (2) Eq. (A.45).
- (3) The definition of SPSP gyro norm [161].
- (4) Riemannian parallel transport is norm-preserving [69, Def. 3.1].
- (5) Eq. (A.49).
- (6) Thm. 113.

□

Remark 184. We make the following remarks regarding the gyro MLR and our MLR on the SPSP manifold.

- (1) Eq. (A.49) indicates that when generating $\tilde{A}$ in our RMLR by parallel transporting a tangent vector $A \in T_{\tilde{I}_{p,n}} \mathcal{S}_{n,p}^+$, $\tilde{A}$ is the initial velocity of W in Eq. (A.44).

- (2) Putting the pseudo-gyrodistance and Riemannian margin distance into Eq. (4.18) yields the gyro MLR and our Riemannian MLR. Therefore, Thm. 183 indicates the equivalence of the gyro MLR with our RMLR on the SPSP manifold.
- (3) As a metric g is required to induce gyro-structures, the metric g in gyro SPSP MLR is confined to AIM, LEM, and LCM. However, our SPSP MLR can be defined on the product space of the Grassmannian and SPD manifold under other metrics, such as BWM and PEM, as our framework does not require gyro-structures.

A.4.1.3 Theories on the Deformed Metrics

Limiting cases of the deformed metrics. Thanwerdas and Pennec [192] generalized (α, β) -AIM to a three-parameter family of metrics by power deformation, *i.e.*, (θ, α, β) -AIM. The family of (θ, α, β) -AIM comprises (α, β) -AIM for $\theta = 1$ and approaches (α, β) -LEM with $\theta \rightarrow 0$ [192].

As reviewed in Sec. 3.2.5.1, LCM and (α, β) -LEM are extended to power-deformed metrics, denoted as (θ, α, β) -LEM and θ -LCM. Thm. 67 shows that (θ, α, β) -LEM is equal to (α, β) -LEM, and θ -LCM interpolates between $\tilde{g}$ -LEM ($\theta \rightarrow 0$) and LCM ($\theta = 1$), with $\tilde{g}$ -LEM defined as

$$\langle V_1, V_2 \rangle_P = \frac{1}{2} \langle \tilde{V}_1, \tilde{V}_2 \rangle - \frac{1}{4} \langle \mathbb{D}(\tilde{V}_1), \mathbb{D}(\tilde{V}_2) \rangle, \forall V_i \in T_P \mathcal{S}_{++}^n, \quad (\text{A.51})$$

where $\tilde{V}_i = \log_{*,P}(V_i)$ with $\log_{*,P}$ as the differential map of matrix logarithm, and $\mathbb{D}(\tilde{V}_i)$ is a diagonal matrix consisting of the diagonal elements of $\tilde{V}_i$.

Thanwerdas and Pennec [194] identified the Alpha-Procrustes metric [151] with power-deformed BWM, denoted as 2θ -BWM. Similarly, 2θ -BWM becomes BWM with $\theta = 0.5$ [194]. We further show the limiting case of 2θ -BWM under $\theta \rightarrow 0$.

Proposition 185. *2θ -BWM tends to $(\frac{1}{4}, 0)$ -LEM as $\theta \rightarrow 0$.*

Before starting the proof, we first recall a well-known property of deformed metrics [194].

Lemma 186. *Let $\frac{1}{\theta^2} \phi_\theta^* g$ be the deformed metric on SPD manifolds pulled back from g by the matrix power ϕ_θ and scaled by $\frac{1}{\theta^2}$. Then, as θ tends to 0, for all $P \in \mathcal{S}_{++}^n$*

and all $V \in T_P \mathcal{S}_{++}^n$, we have

$$(\frac{1}{\theta^2} \phi_\theta^* g)_P(V, V) \rightarrow g_I(\log_{*,P}(V), \log_{*,P}(V)). \quad (\text{A.52})$$

Now, we present our proof for the limiting cases of deformed metrics.

Proof of Thm. 185. First, we have

$$g_I^{\text{BWM}}(V, V) = \frac{1}{4} \langle V, V \rangle. \quad (\text{A.53})$$

By Thm. 186, we have the following:

$$\begin{aligned} g_P^{2\theta\text{-BWM}}(V, V) &\xrightarrow{\theta \rightarrow 0} g_I^{\text{BWM}}(\log_{*,P}(V), \log_{*,P}(V)) \\ &= \frac{1}{4} \langle \log_{*,P}(V), \log_{*,P}(V) \rangle \\ &= g_P^{(\frac{1}{4}, 0)\text{-LEM}}(V, V). \end{aligned} \quad (\text{A.54})$$

□

Proof of the properties of the deformed metrics (Tab. 4.2). In this subsection, we prove the properties presented in Tab. 4.2. We first present a useful lemma and then present our detailed proof. This lemma will be useful in the proof of our SPD MLRs as well.

Lemma 187. *Given a Riemannian manifold $(\mathcal{M}, g)$ and a positive real scalar $a > 0$, the scaled metric ag on $\mathcal{M}$ has the same Riemannian logarithmic maps, exponential maps, and parallel transport as g .*

Proof. Since the Christoffel symbols of ag are identical to those of g , the geodesics and parallel transport under both ag and g remain unchanged. The equivalence of geodesics implies that the Riemannian exponential maps are the same for ag and g . As the inverse of the Riemannian exponential maps, the Riemannian logarithmic maps under ag and g are also identical. □

According to Thm. 187, geodesic completeness is independent of the scaling factor $a > 0$. By the definition of $O(n)$ -, left-, right-, and bi-invariance, these invariant properties are also independent of the scaling factor $a > 0$. Without loss of generality, we will omit the scaling factor in the following proof.

Proof. First, we prove $O(n)$ -invariance of (θ, α, β) -LEM, (θ, α, β) -EM, (θ, α, β) -AIM, and 2θ -BWM. Since the differential of ϕ_θ is $O(n)$ -equivariant, and (α, β) -LEM, (α, β) -EM, (α, β) -AIM, and BWM are $O(n)$ -invariant [196], $O(n)$ -invariance is thus acquired.

Next, we focus on geodesic completeness. It can be easily proven that Riemannian isometries preserve geodesic completeness. On the other hand, (α, β) -LEM, (α, β) -AIM, and LCM are geodesically complete [196, 137]. As a direct corollary, geodesic completeness can be obtained since ϕ_θ is a Riemannian isometry.

Finally, we deal with Lie group invariance. Similarly, it can be readily proved that Lie group invariance is preserved under isometries. LCM, LEM, and (α, β) -AIM are Lie group bi-invariant [137], bi-invariant [9], and left-invariant [195]. As an isometric pullback metric from the standard LEM [196], (α, β) -LEM is, therefore, Lie group bi-invariant. As pullback metrics, (θ, α, β) -LEM, (θ, α, β) -AIM, and θ -LCM are therefore bi-invariant, left-invariant, and bi-invariant, respectively. $\square$

A.4.1.4 Computational Details on the SPD MLR under Power-Deformed BWM

Matrix square roots in the SPD MLR under power-deformed BWM. In the case of MLRs induced by 2θ -BWM, computing square roots like $(BA)^{\frac{1}{2}}$ and $(AB)^{\frac{1}{2}}$ with $B, A \in \mathcal{S}_{++}^n$ poses a challenge. Eigendecomposition cannot be directly applied since BA and AB are no longer symmetric, let alone positive definite. Instead, we use the following formulas to compute these square roots [151]:

$$(BA)^{\frac{1}{2}} = B^{\frac{1}{2}}(B^{\frac{1}{2}}AB^{\frac{1}{2}})^{\frac{1}{2}}B^{-\frac{1}{2}} \text{ and } (AB)^{\frac{1}{2}} = [(BA)^{\frac{1}{2}}]^{\top}, \quad (\text{A.55})$$

where the involved square roots can be computed using eigendecomposition or singular value decomposition (SVD).

Numerical stability of the SPD MLR under power-deformed BWM. Let us first explain why we abandon parallel transport for the SPD MLR derived from 2θ -BWM. Then, we propose our numerically stable methods for computing the SPD MLR based on 2θ -BWM.

Instability of parallel transport under power-deformed BWM. As discussed in Thm. 114, there are two ways to generate $\tilde{A}$ in SPD MLR: parallel transport and Lie group translation. However, parallel transport under 2θ -BWM can cause numerical problems. Without loss of generality, we focus on the standard BWM because 2θ -BWM is isometric to the BWM.

Although general parallel transport under BWM is obtained by solving an ODE,

transport starting from the identity matrix has a closed-form expression [196]:

$$\Gamma_{I \rightarrow P}(V) = U \left[\sqrt{\frac{\sigma_i + \sigma_j}{2}} [U^\top V U]_{ij} \right] U^\top, \quad (\text{A.56})$$

where $P = U \Sigma U^\top$ is the eigendecomposition of $P \in \mathcal{S}_{++}^n$. The forward computation of Eq. (A.56) is numerically stable. However, its backpropagation requires differentiating through an eigendecomposition, involving the calculation of $1/(\sigma_i - \sigma_j)$ [110, Prop. 2]. When σ_i is close to σ_j , this backpropagation can be problematic.

Numerically stable methods for the SPD MLR under power-deformed BWM. To bypass the instability of parallel transport under BWM, we use Lie group left translation to generate $\tilde{A}$ in MLRs induced by 2θ -BWM. However, there is another problem that could cause instability. The computation of the Riemannian metric of 2θ -BWM requires solving the Lyapunov operator, *i.e.*, $\mathcal{L}_P[V]P + P\mathcal{L}_P[V] = V$. For symmetric matrices, the Lyapunov operator can be obtained by eigendecomposition:

$$\mathcal{L}_P[V] = U \left[\frac{V'_{ij}}{\sigma_i + \sigma_j} \right]_{i,j} U^\top, \quad (\text{A.57})$$

where $V \in \mathcal{S}^n$, $UV'U^\top = V$, and $P = U \Sigma U^\top$ is the eigendecomposition of $P \in \mathcal{S}_{++}^n$. Similar to Eq. (A.56), the backpropagation of Eq. (A.57) requires $1/(\sigma_i - \sigma_j)$, undermining numerical stability.

To remedy this problem, we propose the following formula to stably compute the backpropagation of Eq. (A.57).

Proposition 188. *For all $P \in \mathcal{S}_{++}^n$ and all $V \in \mathcal{S}^n$, we denote the Lyapunov equation as*

$$XP + PX = V, \quad (\text{A.58})$$

where $X = \mathcal{L}_P[V]$. Given the gradient $\frac{\partial L}{\partial X}$ of loss L with respect to X , the backpropagation of the Lyapunov operator can be computed by

$$\frac{\partial L}{\partial V} = \mathcal{L}_P \left[\frac{\partial L}{\partial X} \right], \quad (\text{A.59})$$

$$\frac{\partial L}{\partial P} = -X \mathcal{L}_P \left[\frac{\partial L}{\partial X} \right] - \mathcal{L}_P \left[\frac{\partial L}{\partial X} \right] X, \quad (\text{A.60})$$

where $\mathcal{L}_P[\cdot]$ can be computed by Eq. (A.57).

Proof. Differentiating both sides of Eq. (A.58), we obtain

$$\mathrm{d}XP + X\mathrm{d}P + \mathrm{d}PX + P\mathrm{d}X = \mathrm{d}V, \quad (\text{A.61})$$

$$\implies \mathrm{d}XP + P\mathrm{d}X = \mathrm{d}V - X\mathrm{d}P - \mathrm{d}PX, \quad (\text{A.62})$$

$$\implies \mathrm{d}X = \mathcal{L}_P[\mathrm{d}V - X\mathrm{d}P - \mathrm{d}PX]. \quad (\text{A.63})$$

Besides, easy computations show that

$$\mathcal{L}_P[V] : W = V : \mathcal{L}_P[W], \forall W, V \in \mathcal{S}^n, \quad (\text{A.64})$$

where $\cdot : \cdot$ denotes the standard Frobenius inner product.

Then we have the following:

$$\frac{\partial L}{\partial X} : \mathrm{d}X = \frac{\partial L}{\partial X} : \mathcal{L}_P[\mathrm{d}V - X\mathrm{d}P - \mathrm{d}PX], \quad (\text{A.65})$$

$$\implies \frac{\partial L}{\partial X} : \mathrm{d}X = \mathcal{L}_P\left[\frac{\partial L}{\partial X}\right] : \mathrm{d}V + \left(-X\mathcal{L}_P\left[\frac{\partial L}{\partial X}\right] - \mathcal{L}_P\left[\frac{\partial L}{\partial X}\right]X\right) : \mathrm{d}P. \quad (\text{A.66})$$

□

Remark 189. Eq. (A.57) needs to be computed in the Lyapunov operator's forward and backward processes. Therefore, in the forward process, we can save the intermediate matrices U and K with $K_{i,j} = \left[\frac{1}{\sigma_i + \sigma_j}\right]_{i,j}$, and then use them to compute the backward process efficiently.

A.4. Additional Discussions

Method	Point-to-hyperplane distance	Applied manifolds	Dist
Euclidean MLR	$\frac{ \langle a, x \rangle + b }{\ a\ }$	$\mathbb{R}^n$	Real
Poincaré MLR [76]	$\frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{2\sqrt{-K} \langle -p \oplus_M x, a \rangle }{(1 + K \ -p \oplus_M x\ ^2) \ a\ } \right)$ [76, Thm. 5]	$\mathbb{P}_K^n$	Real
Pseudo-Busemann MLR [162]	$d(x, p) \frac{B^v(-p \oplus_M x)}{\ -p \oplus_M x\ }$ [162, Cor. 4.3]	$\mathbb{P}_K^n$	Pseudo
Lorentz MLR [15]	$\frac{1}{\sqrt{-K}} \left \sinh^{-1} \left(\sqrt{-K} \frac{\langle v, x \rangle_{\mathcal{L}}}{\ v\ _{\mathcal{L}}} \right) \right $ [15, Eq. (44)]	$\mathbb{L}_K^n$	Real
BMLR	$\frac{ -\alpha B^v(x) + b }{\alpha}$	$\mathbb{P}_K^n, \mathbb{L}_K^n$	Real

Table A.17: Comparison of point-to-hyperplane distances. **Real** means the point-to-hyperplane distance is the real distance, obtained by $\inf_{y \in H} d(x, y)$ with H as a hyperplane and d as the geodesic distance. Instead, **Pseudo** means the point-to-hyperplane distance is a surrogate, which only equals the real distance in Euclidean geometry.

A.4.2 Hyperbolic Busemann Neural Networks

A.4.2.1 Comparison with Existing Hyperbolic MLRs

Method	Hyperplane	Formulation	Applied manifolds	Compact params
Euclidean MLR	Euclidean	$\{x \in \mathbb{R}^n \mid \langle a, x \rangle + b = 0\}$ $a \in \mathbb{R}^n, b \in \mathbb{R}$	$\mathbb{R}^n$	✓
Poincaré MLR [76]	Geodesic [76, Def. 3.1]	$\{x \in \mathbb{P}_K^n \mid \langle \text{Log}_p(x), a \rangle_p = 0\}$ $p \in \mathbb{P}_K^n, a \in T_p \mathbb{P}_K^n$	$\mathbb{P}_K^n$	✗
Pseudo-Busemann MLR [162]	Busemann & gyro [162, Def. 4.1]	$\{x \in \mathbb{P}_K^n \mid B^v(-p \oplus_M x) = 0\}$ $v \in \mathbb{S}^{n-1}, p \in \mathbb{P}_K^n$	$\mathbb{P}_K^n$	✗
Lorentz MLR [15]	Ambient Minkowski [15, Eq. (7)]	$\{x \in \mathbb{L}_K^n \mid \langle w, x \rangle_{\mathcal{L}} = 0\}$ $p \in \mathbb{L}_K^n, w \in T_p \mathbb{L}_K^n$	$\mathbb{L}_K^n$	✗
BMLR	Horosphere	$\{x \in \mathcal{H}_K^n \mid -\alpha B^v(x) + b = 0\}$ $\alpha > 0, v \in \mathbb{S}^{n-1}, b \in \mathbb{R}$	$\mathbb{P}_K^n, \mathbb{L}_K^n$	✓

Table A.16: Comparison of hyperplanes. Compact params indicate whether the parameterization requires an additional manifold-valued point.

All the considered hyperbolic MLRs follow a point-to-hyperplane formulation. Therefore, the key difference lies in hyperplanes and point-to-hyperplane distances across methods. In addition to Tab. 5.11, Tabs. A.16 and A.17 further make this comparison.

We draw the following three conclusions.

- (1) **Hyperplanes.** Our BMLR uses Busemann-based horospheres that simultaneously satisfy three desiderata: (i) compact parameterization without a per-class manifold-valued point, whereas other hyperbolic ones³² are over-parameterized; (ii) a natural generalization of Euclidean hyperplanes via horospheres, while the Lorentz MLR relies on the ambient Minkowski space, failing to fully respect the intrinsic geometry; and (iii) applicability across hyperbolic models, whereas the Lorentz MLR is tailored to the Lorentz model.
- (2) **Point-to-Hyperplane Distances.** Although pseudo-Busemann MLR also exploits Busemann functions, it relies on a pseudo point-to-hyperplane distance that only coincides with the real distance in Euclidean geometry. In contrast, our BMLR calculates the real point-to-horosphere distance, ensuring geometric fidelity across hyperbolic models.
- (3) **Batch Efficiency.** Recalling Tab. 5.11, the Poincaré MLR [76] computes logits using $\langle -p_k \oplus_M x, a_k \rangle$ and $\| -p_k \oplus_M x \|^2$. For a batch $X \in \mathbb{R}^{\text{bs} \times n}$ and C classes, evaluating $-p_k \oplus_M X$ for every k yields an intermediate tensor of shape $[\text{bs}, C, n]$, and materializing this tensor can cause GPU out of memory (OOM) when n or C is large. The same limitation holds for the pseudo-Busemann MLR. Consequently, their official implementations compute per class in a for-loop, which is batch inefficient. By contrast, BMLR uses logits $-\alpha_k B^{v_k}(x) + b_k$, whose Busemann term reduces to class-wise inner products $\langle v_k, x \rangle$ (or $\langle v_k, x_s \rangle$). With $X \in \mathbb{R}^{\text{bs} \times n}$ and $V = [v_1, \dots, v_C] \in \mathbb{R}^{n \times C}$, such inner products can be efficiently implemented as a single matrix multiplication XV without any $[\text{bs}, C, n]$ intermediate, yielding high throughput and low memory usage.

A.4.2.2 Busemann Fully Connected Layers and Point-to-Horosphere Distances

An apparently natural attempt to define a hyperbolic FC layer is to replace the LHS of Eq. (5.59) by the *signed point-to-horosphere* distance. The Euclidean hyperplane passing through the origin and orthogonal to $e_k \in \mathbb{R}^m$ is $H_{e_k, \mathbf{0}} = \{y \in \mathbb{R}^m \mid \langle e_k, y \rangle = 0\}$. The corresponding hyperbolic horosphere, following Eq. (5.55), is $H_{e_k, 1, 0} = \{y \in \mathcal{H}_K^m \mid -B^{e_k}(y) = 0\}$. By Eq. (5.57), the signed point-to-horosphere distance to $H_{e_k, 1, 0}$ equals

³²We note that Shimizu et al. [181], Bdeir et al. [15] mitigate this issue through re-parameterization: $a = \text{PT}_{e \rightarrow p}(z)$ and $p = \text{Exp}_e\left(b \frac{z}{\|z\|}\right)$, where $e \in \mathcal{H}_K^n$ denotes the origin, $z \in \mathbb{R}^n$, and $b \in \mathbb{R}$. Nevertheless, the underlying definitions are over-parameterized.

$-B^{e_k}(y)$. Accordingly, we can define an alternative FC mapping $\mathcal{F} : \mathcal{H}_K^n \ni x \mapsto y \in \mathcal{H}_K^m$ via

$$B^{e_k}(y) = \alpha_k B^{v_k}(x) - b_k, \quad k = 1, \dots, m, \quad (\text{A.67})$$

where $u_k(x) = -\alpha_k B^{v_k}(x) + b_k$, and $\alpha_k > 0$, $v_k \in \mathbb{S}^{n-1}$, $b_k \in \mathbb{R}$ are learnable. Although this only differs from Eq. (5.59) on the LHS, the following discussion shows that such a definition is infeasible in general and fails to deliver a valid hyperbolic FC layer.

Poincaré Model. Using Eq. (5.40) with $v = e_k$, Eq. (A.67) becomes

$$\frac{1}{\sqrt{-K}} \log \left(\frac{\|e_k - \sqrt{-K}y\|^2}{1 + K\|y\|^2} \right) = -u_k(x). \quad (\text{A.68})$$

Define $t_k = \exp(-\sqrt{-K}u_k(x)) > 0$. Exponentiating Eq. (A.68) gives

$$t_k (1 + K\|y\|^2) = 1 - 2\sqrt{-K}y_k - K\|y\|^2, \quad k = 1, \dots, m. \quad (\text{A.69})$$

Writing $R = \|y\|^2$, Eq. (A.69) yields an affine expression for each coordinate

$$y_k = c_k + d_k R, \quad c_k = \frac{1 - t_k}{2\sqrt{-K}}, \quad d_k = \frac{\sqrt{-K}}{2} (1 + t_k). \quad (\text{A.70})$$

Imposing $R = \sum_{k=1}^m y_k^2 = \sum_{k=1}^m (c_k + d_k R)^2$ gives a quadratic in R :

$$A_2 R^2 + (A_1 - 1) R + A_0 = 0, \quad (\text{A.71})$$

where, denoting $T = \sum_{k=1}^m t_k$ and $q = \sum_{k=1}^m t_k^2$,

$$A_2 = \frac{-K}{4} (m + 2T + q), \quad A_1 = \frac{m - q}{2}, \quad A_0 = \frac{m - 2T + q}{4(-K)} \geq 0. \quad (\text{A.72})$$

Its discriminant is

$$\begin{aligned}
 \Delta_P &= (A_1 - 1)^2 - 4A_2A_0 \\
 &= \left(\frac{m-q}{2} - 1\right)^2 - 4\left(\frac{-K}{4}(m+2T+q)\right)\left(\frac{m-2T+q}{4(-K)}\right) \\
 &= \left(\frac{m-2-q}{2}\right)^2 - \frac{(m+2T+q)(m-2T+q)}{4} \\
 &= \frac{1}{4}[(m-2-q)^2 - ((m+q)^2 - (2T)^2)] \\
 &= \frac{1}{4}[(m-2-q) - (m+q)][(m-2-q) + (m+q)] + 4T^2 \\
 &= \frac{1}{4}[(-2-2q)(2m-2) + 4T^2] \\
 &= T^2 - (m-1)(1+q).
 \end{aligned} \tag{A.73}$$

A real solution R exists only if $\Delta_P \geq 0$. In addition, feasibility requires $0 \leq R < -1/K$ so that $y \in \mathbb{P}_K^m$. Such conditions can fail for generic $\{u_k(x)\}$, in which case no $y \in \mathbb{P}_K^m$ satisfies Eq. (A.67).

Lorentz Model. Using Eq. (5.41) with $v = e_k$, Eq. (A.67) becomes

$$\frac{1}{\sqrt{-K}} \log \left(\sqrt{-K} (y_t - (y_s)_k) \right) = -u_k(x). \tag{A.74}$$

Define $t_k = \exp(-\sqrt{-K}u_k(x)) > 0$ and $t = (t_1, \dots, t_m)^\top$. Denoting $T = \sum_{k=1}^m t_k$ and $q = \sum_{k=1}^m t_k^2$, we obtain from Eq. (A.74),

$$(y_s)_k = y_t - \frac{t_k}{\sqrt{-K}}, \quad k = 1, \dots, m, \quad \implies \quad y_s = y_t \mathbf{1} - \frac{1}{\sqrt{-K}} t. \tag{A.75}$$

Enforcing the hyperboloid constraint $\|y_s\|^2 - y_t^2 = 1/K$ yields a quadratic in y_t :

$$(m-1)Ky_t^2 + 2\sqrt{-K}Ty_t - (1+q) = 0. \tag{A.76}$$

The discriminant is

$$\begin{aligned}
 \Delta_L &= \left(2\sqrt{-K}T\right)^2 - 4(m-1)K(-(1+q)) \\
 &= 4(-K)T^2 + 4(m-1)K(1+q) \\
 &= 4(-K)[T^2 - (m-1)(1+q)].
 \end{aligned} \tag{A.77}$$

Hence, a real y_t exists only if $T^2 - (m-1)(1+q) \geq 0$. In addition, feasibility re-

quires $y_t > 0$. Such conditions can fail for generic $\{u_k(x)\}$, where no $y \in \mathbb{L}_K^m$ satisfies Eq. (A.67).

Summary. Equating Busemann coordinates as in Eq. (A.67) requires nontrivial inequalities on the responses $\{u_k(x)\}$. These constraints are not guaranteed during learning, so the system can become infeasible and the output y undefined. This motivates our choice in Eq. (5.59) to use signed point-to-hyperplane distances on the LHS, which admit closed-form solutions that are feasible for all inputs and parameters in both the Poincaré and Lorentz models.

A.4.3 Full-Rank Correlation Networks

A.4.3.1 Connections among FC Layers: Correlation, SPD, Poincaré, and Euclidean

We clarify the correspondence between our FC formulation in Eq. (5.69) and previous FC layers.

SPD Manifold. Nguyen et al. [161, Props. 3.4–3.6] introduced three SPD FC layers based on the gyrovector spaces under LEM, LCM, and AIM, respectively. These gyro SPD FC layers share the same definition as Eq. (5.69), except that their signed distance and v_k are defined by gyrovector spaces.

Poincaré Ball. We show that the Poincaré FC layer $\mathcal{F}(\cdot) : \mathbb{P}_K^n \rightarrow \mathbb{P}_K^m$ reviewed in Sec. A.2.4 is also defined as our correlation FC layer in Thm. 141.

We define the zero vector $\mathbf{0} \in \mathbb{P}_K^n$ as the Poincaré origin, as it is the identity element of the Poincaré gyrovector space [76]. Obviously, $\{e_k\}_{k=1}^m$ is the orthogonal basis over $T_{\mathbf{0}}\mathbb{P}_K^m$, where $e_k = (\delta_{ik})_{i=1}^m$. Corresponding to Eq. (5.69), we have

$$\text{sign}(\langle \text{Log}_{\mathbf{0}}(y), e_k \rangle) d(y, H_{e_k, \mathbf{0}}) = v_k(x). \quad (\text{A.78})$$

Compared with Shimizu et al. [181, Eq. (56)], we only need to show the LHS. The sign can be calculated as

$$\begin{aligned} \text{sign}(\langle \text{Log}_{\mathbf{0}}(y), e_k \rangle_{\mathbf{0}}) &\stackrel{(1)}{=} \text{sign} \left(4 \left\langle \tanh^{-1} \left(\sqrt{-K} \|y\| \right) \frac{y}{\sqrt{-K} \|y\|}, e_k \right\rangle \right) \\ &\stackrel{(2)}{=} \text{sign}(\langle y, e_k \rangle) \\ &= \text{sign}(y_k). \end{aligned} \quad (\text{A.79})$$

The above follows from the following.

$$(1) \lambda_{\mathbf{0}}^K = \frac{2}{1+K\|\mathbf{0}\|^2} = 2 \text{ and } \text{Log}_{\mathbf{0}}(y) = \tanh^{-1}(\sqrt{-K}\|y\|) \frac{y}{\sqrt{-K}\|y\|}.$$

$$(2) \tanh^{-1}(a) > 0, \iff a > 0.$$

Therefore, the LHS of Eq. (A.78) is simplified as

$$\begin{aligned} \text{sign}(\langle \text{Log}_{\mathbf{0}}(y), e_k \rangle) d(y, H_{e_k, \mathbf{0}}) &= \text{sign}(y_k) d(y, H_{e_k, \mathbf{0}}) \\ &\stackrel{(1)}{=} \text{sign}(y_k) \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{2\sqrt{-K}|y_k|}{1 + K\|y\|^2} \right) \\ &\stackrel{(2)}{=} \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{2\sqrt{-K}y_k}{1 + K\|y\|^2} \right). \end{aligned} \quad (\text{A.80})$$

The above comes from the following.

$$(1) \text{ Ganea et al. [76, Thm. 5].}$$

$$(2) 1 + K\|y\|^2 > 0 \text{ by definition and } \text{sign}(a) \sinh^{-1}(|a|) = \sinh^{-1}(a), \forall a \in \mathbb{R}.$$

The last equation in Eq. (A.80) is the LHS of Shimizu et al. [181, Eq. (56)], indicating the equality.

Euclidean Space. We show that the Euclidean FC layer $\mathcal{F}(\cdot) : \mathbb{R}^n \ni x \rightarrow y = Ax + b \in \mathbb{R}^m$ can also be defined as our correlation FC layer in Thm. 141.

In Euclidean space $\mathbb{R}^n$, the zero vector $\mathbf{0} \in \mathbb{R}^n$ is the origin, and $\{e_k\}_{k=1}^m$ is the orthogonal basis over $T_{\mathbf{0}}\mathbb{R}^m \cong \mathbb{R}^m$. Then, the RHS of Eq. (5.69) becomes

$$\begin{aligned} v_k(x) &= \langle x - p_k, a_k \rangle \\ &\stackrel{(1)}{=} \langle x, z_k \rangle - \gamma_k \|z_k\|. \end{aligned} \quad (\text{A.81})$$

where (1) comes from $\text{Exp}_{\mathbf{0}}(\gamma_k[z_k]) = \gamma_k[z_k]$ and $\text{PT}_{\mathbf{0} \rightarrow p_k}(z_k) = z_k$. The above takes the form of $\langle x, a_k \rangle + b_k$.

On the other hand, the LHS of Eq. (5.69) becomes

$$\begin{aligned} \text{sign}(\langle \text{Log}_{\mathbf{0}}(y), e_k \rangle) d(y, H_{e_k, \mathbf{0}}) &\stackrel{(1)}{=} \text{sign}(y_k) d(y, H_{e_k, \mathbf{0}}) \\ &\stackrel{(2)}{=} y_k. \end{aligned} \quad (\text{A.82})$$

The above comes from the following.

$$(1) \text{Log}_{\mathbf{0}}(y) = y \text{ and } \langle y, e_k \rangle = y_k.$$

$$(2) d(y, H_{e_k, \mathbf{0}}) = \frac{|\langle y, e_k \rangle|}{\|e_k\|} = |y_k|.$$

A.4.3.2 Log-Euclidean Layers under Product Geometry

We first review some basic facts of the product geometry, and then discuss the Log-Euclidean correlation MLR and FC layer under the product geometry.

Product of Correlation Manifolds. Given a manifold $(\mathcal{M}, g)$, the n -fold product is $(\mathcal{M}^n, g) = \prod_{i=1}^n (\mathcal{M}, g)$. Each point and tangent vector over $\mathcal{M}^n$ are

$$\mathcal{M}^n \ni P = (P_1 \in \mathcal{M}, \dots, P_n \in \mathcal{M}), \quad (\text{A.83})$$

$$T_P \mathcal{M}^n \ni V = (V_1 \in T_{P_1} \mathcal{M}, \dots, V_n \in T_{P_n} \mathcal{M}). \quad (\text{A.84})$$

The product metric is

$$\langle V, W \rangle_P = \sum_{i=1}^n \langle V_i, W_i \rangle_{P_i}, \quad \forall V, W \in T_P \mathcal{M}^n. \quad (\text{A.85})$$

Correlation MLR. Following Thm. 138, the logit for the k -th class and the input $\mathbf{X} = \{X_l \in \text{Cor}^+(n)\}_{l=1}^c \in (\text{Cor}^+(n))^c$ is

$$\begin{aligned} v_k(\mathbf{X}) &= \sum_{l=1}^c v_{kl}(X_l; Z_{kl}, \gamma_{kl}) \\ &= \sum_{l=1}^c (\langle \phi_n(X_l), (\phi_n)_{*, I_n}(Z_{kl}) \rangle - \gamma_{kl} \|(\phi_n)_{*, I_n}(Z_{kl})\|), \end{aligned} \quad (\text{A.86})$$

where $d_n = n(n-1)/2$, $\phi_n : \text{Cor}^+(n) \rightarrow \mathbb{R}^{d_n}$ is the diffeomorphism of the selected Log-Euclidean geometry, $\mathbf{I} = (I_n, \dots, I_n)$, $Z_k = (Z_{k1}, \dots, Z_{kc}) \in T_{\mathbf{I}}(\text{Cor}^+(n))^c$, $Z_{kl} \in T_{I_n} \text{Cor}^+(n)$, and $\gamma_{kl} \in \mathbb{R}$. Each v_{kl} is the response of the l -th component given by Thm. 138, with the unit direction $[Z_{kl}] = Z_{kl} / \|Z_{kl}\|_{I_n}$.

Correlation FC Layer. Following Thm. 201 and Eq. (A.86), the FC layer $\mathcal{F}(\cdot) : (\text{Cor}^+(n))^c \rightarrow \text{Cor}^+(m)$ for the input $\mathbf{X}$ is

$$Y = \phi_m^{-1} \left(\sum_{i=1}^{d_m} \sum_{l=1}^c v_{il}(X_l; Z_{il}, \gamma_{il}) e_i \right), \quad (\text{A.87})$$

where $d_m = m(m-1)/2$, $\phi_m : \text{Cor}^+(m) \rightarrow \mathbb{R}^{d_m}$ is the output diffeomorphism, $Z_{il} \in T_{I_n} \text{Cor}^+(n) \cong \text{Hol}(n)$, and $\gamma_{il} \in \mathbb{R}$.

Eq. (A.87) implies that $\mathcal{F}(\cdot) : (\text{Cor}^+(n))^c \rightarrow \text{Cor}^+(m)$ differs from $\mathcal{F}(\cdot) : \text{Cor}^+(n) \rightarrow \text{Cor}^+(m)$ only in each scalar response v_i , where the former is a summation over the c input components. For example, considering the FC layer $\mathcal{F}(\cdot) : (\text{Cor}^+(n))^c \rightarrow \text{Cor}^+(m)$

under ECM, its matrix-coordinate response v_{ij} for the input $\mathbf{C} = \{C_l \in \text{Cor}^+(n)\}_{l=1}^c$ is

$$v_{ij}(\mathbf{C}) = \sum_{l=1}^c v_{ijl}^{\text{EC}}(C_l; Z_{ijl}, \gamma_{ijl}), \quad (\text{A.88})$$

where $Z_{ijl} \in \text{Hol}(n)$ and $\gamma_{ijl} \in \mathbb{R}$, for $i, j = 1, \dots, m$ with $i > j$, and $1 \leq l \leq c$.

A.4.4 Adaptive Log-Euclidean Metrics

A.4.4.1 Well-definedness of General Matrix Logarithm

Eq. (6.12) does not explicitly specify the correspondence between eigenvalues and diagonal logarithms. Here, we present a detailed explanation. In implementations such as PyTorch or MATLAB, eigendecomposition routines return eigenpairs in a prescribed order.

We rewrite the eigendecomposition as $S = \sum \sigma_i E_i$, where $E_i = u_i u_i^\top$ and u_i is the corresponding eigenvector in U . Let S be an $n \times n$ SPD matrix, and let P_n be the permutation group on $\{1, \dots, n\}$. Changing the order of $\{1, \dots, n\}$ can be viewed as a permutation, so we use $\pi \in P_n$ to represent the corresponding changed order.

Assume that the eigenvalues σ_i are sorted in ascending order, *i.e.*, $\sigma_1 \leq \dots \leq \sigma_n$. We use “the i -th eigenvalue” to refer to the eigenvalue in the i -th pair of the ordered eigenpair sequence $(\sigma_1, u_1), \dots, (\sigma_n, u_n)$. Since each eigenvector u_i is unique, it is safe to say the eigenvalues are ordered, and the i -th eigenvalue/eigenvector pair is unique.

Let $\log_\alpha(\Sigma)$ denote applying the scalar logarithm $\log_{\alpha_i}$ to the i -th eigenvalue σ_i . Then $\log_\alpha(S)$ is rewritten as $\log_\alpha(S) = \sum \log_{\alpha_i}(\sigma_i) E_i$, where $S = \sum \sigma_i E_i$. In this way, $\log_\alpha$ is clearly well-defined. By definition, we can observe that the output of $\log_\alpha$ does not depend on the order in eigendecomposition.

Suppose there are two eigendecompositions with different orders, *i.e.*, $S = U \Sigma U^\top = \tilde{U} \tilde{\Sigma} \tilde{U}^\top$, where $\tilde{U}$ and $\tilde{\Sigma}$ are rearrangements of U and Σ . There exists a $\pi \in P_n$ such that, for each j , there is a unique i satisfying $\tilde{u}_j = u_{\pi(i)}$ and $\tilde{\sigma}_j = \sigma_{\pi(i)}$. We then have $\sum \log_{\alpha_i}(\sigma_i) E_i$ for $S = U \Sigma U^\top$ and $\sum \log_{\alpha_{\pi(i)}}(\sigma_{\pi(i)}) E_{\pi(i)}$ for $S = \tilde{U} \tilde{\Sigma} \tilde{U}^\top$, which indicates that the two eigendecompositions are equivalent.

A.4.5 Product Cholesky Metrics

A.4.5.1 Riemannian Operators under $(\theta, \mathbb{M})$ -DBWM

We first define a map $\phi_\theta : \mathcal{L}_{++}^n \rightarrow \mathcal{L}_{++}^n$ as

$$\phi_\theta(L) = \lfloor L \rfloor + \mathbb{L}^\theta, \forall L \in \mathcal{L}_{++}^n. \quad (\text{A.89})$$

Its differential at $L \in \mathcal{L}_{++}^n$ is given by

$$(\phi_\theta)_{*,L}(X) = \lfloor X \rfloor + \theta \mathbb{L}^{\theta-1} \mathbb{X}, \quad \forall X \in T_L \mathcal{L}_{++}^n. \quad (\text{A.90})$$

Let $g^{\mathbb{M}\text{-DBW}}$ and $g^{(\theta, \mathbb{M})\text{-DBW}}$ be $\mathbb{M}$ -DBWM and $(\theta, \mathbb{M})$ -DBWM, respectively. Since constant scaling of a Riemannian metric preserves the Christoffel symbols, Riemannian operators such as the Riemannian logarithm, exponential map, and parallel transport under $g^{(\theta, \mathbb{M})\text{-DBW}}$ are the same as those under the pullback metric $\phi_\theta^* g^{\mathbb{M}\text{-DBW}}$. Following Thm. 33, these Riemannian operators under $(\theta, \mathbb{M})$ -DBWM can be obtained by $\phi_\theta^* g^{\mathbb{M}\text{-DBW}}$. Besides, as constant scaling does not affect the weighted Fréchet mean (WFM), the WFM under $(\theta, \mathbb{M})$ -DBWM is the same as that under $\phi_\theta^* g^{\mathbb{M}\text{-DBW}}$. The latter can be calculated by the properties of isometries presented in Thm. 33. Therefore, by the properties of isometry in Thm. 33 and Eqs. (A.89) and (A.90), we can obtain all the Riemannian operators.

Appendix B

Proofs

B.1 Mathematical Background

B.1.1 Proof of Thm. 57

Proof. The gyration-preservation property is shown by Ungar [200, Eq. (6.323)]. The proofs for the gyro identity and gyroinverse follow directly from the isomorphism and the uniqueness of inverse and identity [200, Thm. 2.10]. $\square$

B.2 Lie Group Batch Normalization

B.2.1 Proof of Thm. 61

Proof. **Case (1).** The MLE of M is

$$\begin{aligned} M_{\text{MLE}} &= \operatorname{argmax} \log(k(v)) - \sum_{i=1}^N \frac{d(P_i, M)^2}{2v^2} \\ &= \operatorname{argmin} \sum_{i=1}^N d(P_i, M)^2. \end{aligned} \tag{B.1}$$

Case (2). We denote $Y = L_B(X)$, and p_X and p_Y as the density of X and Y ,

respectively. The density of Y is

$$\begin{aligned}
 p_Y(Q) &\stackrel{(1)}{=} p_X(L_{\ominus B}(Q)) \\
 &= k(\sigma) \exp\left(-\frac{d(L_{\ominus B}(Q), M)^2}{2\sigma^2}\right) \\
 &\stackrel{(2)}{=} k(\sigma) \exp\left(-\frac{d(Q, L_B(M))^2}{2\sigma^2}\right).
 \end{aligned} \tag{B.2}$$

The above comes from:

- (1) Pennec [170, Thm. 7];
- (2) The isometry of the left translation.

□

B.2.2 Proof of Thm. 62

Proof. The isometry of L_B directly implies the homogeneity of the sample mean. Now let us focus on Eq. (3.14). We have the following:

$$\begin{aligned}
 \sum_{i=1}^N w_i d^2(\phi_s(P_i), E) &= \sum_{i=1}^N w_i \|s \operatorname{Log}_E P_i\|_E^2 \\
 &= s^2 \sum_{i=1}^N w_i \|\operatorname{Log}_E P_i\|_E^2 \\
 &= s^2 \sum_{i=1}^N w_i d^2(P_i, E),
 \end{aligned} \tag{B.3}$$

where $\|\cdot\|_E$ is the norm on $T_E \mathcal{M}$.

□

B.2.3 Proof of Thm. 65

Proof. As the right-invariant metric has properties analogous to those of the left-invariant metric, this proof follows the same logic as the two proofs above.

Gaussian homogeneity. We denote $Y = R_B(X)$, and p_X and p_Y as the density of X and Y , respectively. The density of Y is

$$\begin{aligned}
 p_Y(Q) &\stackrel{(1)}{=} p_X(R_{\ominus B}(Q)) \\
 &= k(\sigma) \exp\left(-\frac{d(R_{\ominus B}(Q), M)^2}{2\sigma^2}\right) \\
 &\stackrel{(2)}{=} k(\sigma) \exp\left(-\frac{d(Q, R_B(M))^2}{2\sigma^2}\right).
 \end{aligned} \tag{B.4}$$

The above comes from:

- (1) Pennec [170, Thm. 13];
- (2) The isometry of the right translation.

Sample mean homogeneity. This is a direct corollary of the isometry of right translation. $\square$

B.2.4 Proof of Thm. 66

Proof. As $\mathbb{R}^n$ is an abelian group and the Euclidean inner product is bi-invariant, we focus on left translation in the following. The core of this proof lies in the fact that on $\mathbb{R}^n$, (1) the Fréchet mean and variance are reduced to the familiar Euclidean statistics. (2) the calculation of the running mean becomes the weighted arithmetic mean. (3) Eqs. (3.8) to (3.10) become Eq. (3.1). We prove these three points one by one.

As stated by Lou et al. [142, Prop. G.1 and Cor. G.2], from the view of the product manifold, the element-wise Fréchet mean and variance on $\mathbb{R}^n$ are equivalent to the vector-valued Euclidean mean and variance.

Besides, by an argument similar to that of Lou et al. [142, Prop. G.1], the weighted Fréchet mean on $\mathbb{R}^n$ is simplified to the weighted arithmetic average. Therefore, on $\mathbb{R}^n$, the calculation of running statistics in our Alg. 1 becomes the familiar moving average.

Thirdly, on $\mathbb{R}^n$, we know that $L_x(y) = x + y$, $\text{Exp}_x v = x + v$, $\text{Log}_x y = y - x$, and the neutral element is 0. Since statistics, as well as the Euclidean BN, are calculated element-wise, it suffices to consider a single coordinate, *i.e.*, $\mathbb{R}$. For a batch of activations $\{x_i\}_{i=1}^N \subset \mathbb{R}$ with batch mean μ_b and batch variance v_b^2 , Eqs. (3.8) to (3.10) are rewritten as follows:

$$L_\beta \left(\text{Exp}_0 \left[\frac{\gamma}{\sqrt{v_b^2 + \epsilon}} \text{Log}_0(L_{-\mu_b}(x_i)) \right] \right) = \gamma \frac{x_i - \mu_b}{\sqrt{v_b^2 + \epsilon}} + \beta. \quad (\text{B.5})$$

The above equation is the exact core computation of the standard Euclidean BN. $\square$

B.2.5 Proof of Thm. 67

Proof. We first prove the case of (θ, α, β) -LEM, and then proceed to the case of θ -LCM.

(θ, α, β) -LEM. For clarity, we denote the metric tensor of (θ, α, β) -LEM as

$$g^{(\theta, \alpha, \beta)\text{-LE}} = \frac{1}{\theta^2} P_\theta^* g^{(\alpha, \beta)\text{-LE}}, \quad (\text{B.6})$$

where $g^{(\alpha, \beta)\text{-LE}}$ is the metric tensor of (α, β) -LEM. Let $P \in \mathcal{S}_{++}^n$ and $V, W \in T_P \mathcal{S}_{++}^n$.

Then we have

$$\begin{aligned}
 g_P^{(\theta, \alpha, \beta)\text{-LE}}(V, W) &= \frac{1}{\theta^2} g_{P_\theta(P)}^{(\alpha, \beta)\text{-LE}}((P_\theta)_{*,P}(V), (P_\theta)_{*,P}(W)) \\
 &= \frac{1}{\theta^2} \langle (\log \circ P_\theta)_{*,P}(V), (\log \circ P_\theta)_{*,P}(W) \rangle^{(\alpha, \beta)} \\
 &= \langle \log_{*,P}(V), \log_{*,P}(W) \rangle^{(\alpha, \beta)} \\
 &= g_P^{(\alpha, \beta)\text{-LE}}(V, W).
 \end{aligned} \tag{B.7}$$

θ -LCM. Let us first review a well-known fact of deformed metrics [194]. Let $\tilde{g} = \frac{1}{\theta^2} P_\theta^* g$ be the power-deformed metric on SPD. Then when θ tends to 0, for all $P \in \mathcal{S}_{++}^n$ and all $V \in T_P \mathcal{S}_{++}^n$, we have

$$\tilde{g}_P(V, V) \rightarrow g_P(\log_{*,P}(V), \log_{*,P}(V)). \tag{B.8}$$

By Eq. (B.8), we can readily obtain the results. $\square$

B.2.6 Proof of Thm. 68

Proof. (α, β) -AIM is left-invariant [195]. As the pullback of (α, β) -AIM, (θ, α, β) -AIM is left-invariant as well. Besides, Thm. 147 shows that LCM is the pullback metric from the Euclidean space of LT^n . Therefore, θ -LCM is bi-invariant. $\square$

B.2.7 Proof of Thm. 69

Proof. In the following, we denote the Riemannian operators under the left-invariant metric, *i.e.*, AIM, by d^L , Log^L , and Exp^L . Note that the Cholesky decomposition pulls back the group operation of matrix product from the Cholesky manifold $\mathcal{L}_{++}^n$ [195]. For simplicity, we abbreviate $\oplus^{\text{LieAI}}$ as $\oplus$.

The differential maps of the Cholesky decomposition and its inverse are reviewed in Eq. (2.92). Following the notation in this theorem, we further denote $X \in T_L \mathcal{L}_{++}^n$. Specifically, for the differential map at I , we have

$$\text{Chol}_{*,I}(V) = (V)_{\frac{1}{2}}, \forall V \in T_I \mathcal{S}_{++}^n. \tag{B.9}$$

$$(\text{Chol}^{-1})_{*,I}(X) = (X)_{\text{sym}+}, \forall X \in T_I \mathcal{L}_{++}^n. \tag{B.10}$$

Denoting $\tilde{L}$ and $\tilde{R}$ as the group translation on the Cholesky manifold $\mathcal{L}_{++}^n$, we have the

following w.r.t. the differential maps of left and right translation:

$$((L_P)_{*,\ominus P})^{-1} \stackrel{(1)}{=} \left((\text{Chol}^{-1})_{*,I} (\tilde{L}_L)_{*,L^{-1}} \circ \text{Chol}_{*,\ominus P} \right)^{-1} \quad (\text{B.11})$$

$$= (\text{Chol}_{*,\ominus P})^{-1} \left((\tilde{L}_L)_{*,L^{-1}} \right)^{-1} \circ \text{Chol}_{*,I}, \quad (\text{B.12})$$

$$(\text{R}_{\ominus P})_{*,P} \stackrel{(2)}{=} (\text{Chol}^{-1})_{*,I} \circ (\tilde{\text{R}}_{L^{-1}})_{*,L} \circ \text{Chol}_{*,P}. \quad (\text{B.13})$$

The above derivation comes from the following:

$$(1) \quad L_P = \text{Chol}^{-1} \circ \tilde{L}_L \circ \text{Chol};$$

$$(2) \quad \text{R}_{\ominus P} = \text{Chol}^{-1} \circ \tilde{\text{R}}_{L^{-1}} \circ \text{Chol}.$$

Riemannian metric. For the differential of right translation, we have the following:

$$\begin{aligned} (\text{R}_{\ominus P})_{*,P}(V) &= (\text{Chol}^{-1})_{*,I} \circ (\tilde{\text{R}}_{L^{-1}})_{*,L} \circ \text{Chol}_{*,P}(V) \\ &= \left(L(L^{-1}VL^{-\top})_{\frac{1}{2}} L^{-1} \right)_{\text{sym}+}. \end{aligned} \quad (\text{B.14})$$

By Eq. (B.14), one can obtain the expression for the Riemannian metric tensor.

Riemannian geodesic and exponential map. According to Zacur et al. [228], we have the following for the operators between left- and right-invariant metrics:

$$\text{Exp}_P(V) = \ominus \left\{ \text{Exp}_{\ominus P}^L \left(-((L_P)_{*,\ominus P})^{-1} \circ (\text{R}_{\ominus P})_{*,P}(V) \right) \right\}, \quad (\text{B.15})$$

$$\text{d}(P, Q) = \text{d}^L(\ominus P, \ominus Q). \quad (\text{B.16})$$

Putting the AIM-based geodesic distance into the RHS of Eq. (B.16), one can obtain the geodesic distance under CRIM.

Now, we simplify Eq. (B.15). Putting Eqs. (B.12) and (B.13) into Eq. (B.15), we have the following:

$$\begin{aligned} \text{Exp}_P(V) &= \ominus \left\{ \text{Exp}_{\ominus P}^L \left(-(\text{Chol}_{*,\ominus P})^{-1} \circ \left((\tilde{L}_L)_{*,L^{-1}} \right)^{-1} \circ (\tilde{\text{R}}_{L^{-1}})_{*,L} \circ \text{Chol}_{*,P}(V) \right) \right\} \\ &= \ominus \left\{ \text{Exp}_{\ominus P}^L \left(-(\text{Chol}_{*,\ominus P})^{-1} [L^{-1} \text{Chol}_{*,P}(V) L^{-1}] \right) \right\} \\ &\stackrel{(1)}{=} \ominus \left\{ \text{Exp}_{\ominus P}^L \left(-(\text{Chol}_{*,\ominus P})^{-1} \left[(L^{-1}VL^{-\top})_{\frac{1}{2}} L^{-1} \right] \right) \right\} \\ &\stackrel{(2)}{=} \ominus \left\{ \text{Exp}_{\ominus P}^L \left(- \left((L^{-1}VL^{-\top})_{\frac{1}{2}} L^{-1} L^{-\top} \right)_{\text{sym}+} \right) \right\}. \end{aligned} \quad (\text{B.17})$$

The above comes from the following:

- (1) The first identity follows from Eq. (2.92);
- (2) The second identity follows from Eq. (2.92).

Riemannian logarithm. From the second equality in Eq. (B.17), we have the following:

$$\begin{aligned}
 \text{Log}_P(Q) &= -\text{Chol}_{*,P}^{-1} \left\{ L \text{Chol}_{*,\ominus P} (\text{Log}_{\ominus P}^L (\ominus Q)) L \right\} \\
 &\stackrel{(1)}{=} -\text{Chol}_{*,P}^{-1} \left\{ \left(L \tilde{V} L^\top \right)_{\frac{1}{2}} L \right\} \\
 &= - \left(L L^\top \left(L \tilde{V} L^\top \right)_{\frac{1}{2}}^\top \right)_{\text{sym}+}
 \end{aligned} \tag{B.18}$$

The above comes from the following:

- (1) $\text{Chol}_{*,\ominus P}(V) = L^{-1} (L V L^\top)_{\frac{1}{2}}, \forall V \in T_{\ominus P} \mathcal{S}_{++}^n$.

□

B.2.8 Proof of Thm. 70

Proof. Completeness. Eq. (3.21) indicates that Exp_I is defined over the whole $T_I \mathcal{S}_{++}^n$. Besides, the SPD manifold is connected [171]. By Lee [131, Cor. 6.20], CRIM is complete.

Geodesic. For simplicity, we abbreviate $\oplus^{\text{LieAI}}$ as $\oplus$. The geodesic connecting P and Q can be obtained as follows:

$$\begin{aligned}
 \gamma(t; P, Q) &= \text{Exp}_P (t \text{Log}_P(Q)) \\
 &\stackrel{(1)}{=} \ominus \left\{ \text{Exp}_{\ominus P}^L \left(-(\text{Chol}_{*,\ominus P})^{-1} [L^{-1} \text{Chol}_{*,P} (t \text{Log}_P(Q)) L^{-1}] \right) \right\} \\
 &\stackrel{(2)}{=} \ominus \left\{ \text{Exp}_{\ominus P}^L (t \text{Log}_{\ominus P}^L (\ominus Q)) \right\} \\
 &= \ominus \left\{ \gamma^{\text{AI}}(t; \tilde{P}, \tilde{Q}) \right\}.
 \end{aligned} \tag{B.19}$$

The above comes from the following:

- (1) The second equality in Eq. (B.17);
- (2) The first equality in Eq. (B.18).

□

B.2.9 Proof of Thm. 72

Proof. Without loss of generality, we focus on the case of the left-invariant metric. The results for the right-invariant metric can be proven similarly.

We denote Eqs. (3.8) to (3.10) on $\mathcal{M}_i, i = 1, 2$ as the mapping $\xi^i(\cdot|M, v^2, B, s)$. Let $\mathcal{B} = \{P_i\}_{i=1}^N$ and $f(\mathcal{B}) = \{f(P_i)\}_{i=1}^N$.

The core of this proof lies in three points:

- (1) The Fréchet mean and variance of $\mathcal{B}$ in $\mathcal{M}_1$ correspond to the counterparts of $f(\mathcal{B})$ in $\mathcal{M}_2$.
- (2) $\xi^1(P_i|M, v^2, B, s)$ in $\mathcal{M}_1$ is equal to $f^{-1}(\xi^2(f(P_i)|f(M), v^2, f(B), s))$.
- (3) The updates of running statistics in $\mathcal{M}_1$ correspond to the counterparts in $\mathcal{M}_2$.

We denote M as the Fréchet mean of $\mathcal{B}$, and v^2 as the Fréchet variance of $\mathcal{B}$. Then, by the isometry of f , the Fréchet mean and variance of $f(\mathcal{B})$ are $f(M)$ and v^2 , respectively.

On $\mathcal{M}_i, i = 1, 2$, we denote $L^i, \oplus^i, \text{Exp}^i, \text{Log}^i$ as the Lie group and Riemannian operators, E^i as the neutral element, and Eq. (3.9) as $\phi_s^i(\cdot)$. With the isometry and Lie group isomorphism of f , we have the following equations:

$$L_{\ominus^1 M}^1 = f^{-1} \circ L_{\ominus^2 f(M)}^2 \circ f, \quad (\text{B.20})$$

$$\phi_s^1 = \text{Exp}_{E^1}^1 [s \text{Log}_{E^1}^1(\cdot)] \quad (\text{B.21})$$

$$= f^{-1} (\text{Exp}_{E^2}^2 [s \text{Log}_{E^2}^2(f(\cdot))]) \quad (\text{B.22})$$

$$= f^{-1} \circ \phi_s^2 \circ f, \quad (\text{B.23})$$

$$L_B^1 = f^{-1} \circ L_{f(B)}^2 \circ f. \quad (\text{B.24})$$

Then we have

$$\xi^1(P_i|M, v^2, B, s) = f^{-1}(\xi^2(f(P_i)|f(M), v^2, f(B), s)) \quad (\text{B.25})$$

Lastly, we show the correspondence between running statistics. Since the Fréchet variance is the same for both $\mathcal{B}$ and $f(\mathcal{B})$, we focus on the running mean. Let M_r and $f(M_r)$ denote the initial values of the running means in $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively, and let WFM^i represent the weighted Fréchet mean in $\mathcal{M}_i$. Then the updated running mean in $\mathcal{M}_1$ is

$$\text{WFM}^1(\{1 - \eta, \eta\}, \{M_r, M\}) = f^{-1}(\text{WFM}^2(\{1 - \eta, \eta\}, \{f(M_r), f(M)\})) \quad (\text{B.26})$$

We can further simplify the above equation as

$$\text{WFM}^1 = f^{-1} \circ \text{WFM}^2 \circ f \quad (\text{B.27})$$

Denoting LieBN^i as the LieBN algorithm on $\mathcal{M}_i$, Eq. (B.25) and Eq. (B.27) imply that

$$\text{LieBN}^1(P_i|B, s, \epsilon, \eta) = f^{-1} [\text{LieBN}^2(f(P_i)|f(B), s, \epsilon, \eta)] . \quad (\text{B.28})$$

□

B.3 Gyrogroup Batch Normalization

B.3.1 Proof of Thm. 74

Proof. By Nguyen and Yang [159, Lem. 2.3], easy computations show that Eq. (3.33) holds for $\text{Gr}(p, n)$ if and only if it holds for $\widetilde{\text{Gr}}(p, n)$. Without loss of generality, we prove the case for the projector perspective.

Given any $P, Q \in \widetilde{\text{Gr}}(p, n)$, Nguyen [157, Def. 3.18] gives the expression for gyration:

$$\text{gyr}[\ominus P, P]Q = F(\ominus P, P)Q(F(\ominus P, P))^{-1}, \quad (\text{B.29})$$

with $F(\ominus P, P)$ defined as

$$F(\ominus P, P) = \exp\left(-\left[\overline{\ominus P \oplus P}, \tilde{I}_{p,n}\right]\right) \exp\left(\left[\overline{\ominus P}, \tilde{I}_{p,n}\right]\right) \exp\left(\left[\overline{P}, \tilde{I}_{p,n}\right]\right), \quad (\text{B.30})$$

where $\overline{(\cdot)} = \text{Log}_{\tilde{I}_{p,n}}(\cdot)$. This equation can be further simplified as

$$\begin{aligned} F(\ominus P, P) &\stackrel{(1)}{=} \exp(0) \exp\left(\left[\overline{\ominus P}, \tilde{I}_{p,n}\right]\right) \exp\left(\left[\overline{P}, \tilde{I}_{p,n}\right]\right) \\ &\stackrel{(2)}{=} \exp\left(\left[\overline{\ominus P}, \tilde{I}_{p,n}\right]\right) \exp\left(\left[\overline{P}, \tilde{I}_{p,n}\right]\right) \\ &\stackrel{(3)}{=} I_n. \end{aligned} \quad (\text{B.31})$$

The above derivation follows from

$$(1) \quad \overline{\ominus P \oplus P} = \overline{\tilde{I}_{p,n}} = \mathbf{0}_{n \times n} \in \mathbb{R}^{n \times n}.$$

$$(2) \quad \exp(0) = I_n.$$

$$(3) \quad \overline{\ominus P} = -\overline{P} \text{ and } \exp\left(\left[-\overline{P}, \tilde{I}_{p,n}\right]\right) = \exp\left(\left[\overline{P}, \tilde{I}_{p,n}\right]\right)^{-1}.$$

Therefore, $\text{gyr}[\ominus P, P]$ is the identity map. $\square$

B.3.2 Proof of Thm. 75

Proof. This theorem follows from Ungar [200, Thms. 2.10–2.11], which presents some useful properties for gyrogroups. We argue that all the properties except $\text{gyr}[a, a] = \text{id}$ are independent of the left reduction law (G4), and are therefore satisfied on pseudo-reductive gyrogroups. All the properties can be proven in the same way as in Ungar [200, Thms. 2.10–2.11]. We summarize the logic in the following:

- left gyroassociativity $\Rightarrow$ Case (1)
- left gyroassociativity + Case (1) $\Rightarrow$ Case (2)
- definition $\Rightarrow$ Case (3)
- left gyroassociativity + Case (1) + Case (3) $\Rightarrow$ Case (4)
- definition $\Rightarrow$ Case (5)
- left gyroassociativity + (G2) + Case (1) + Case (3) + Case (4) + Case (5) $\Rightarrow$ Case (6)
- Case (1)+Case (6) $\Rightarrow$ Case (7)
- left gyroassociativity +Case (3) $\Rightarrow$ Case (8)
- left gyroassociativity + left cancellation in Case (8) $\Rightarrow$ Case (9)
- gyro identity in Case (9) $\Rightarrow$ Case (10)
- Case (10) $\Rightarrow$ Case (11)
- left cancellation in Case (8) + gyro identity in Case (9) $\Rightarrow$ Case (12)
- left cancellation in Case (8) + gyro identity in Case (9) $\Rightarrow$ Case (13)

$\square$

B.3.3 Proof of Thm. 77

Proof.

$$\begin{aligned}
 y &\stackrel{(1)}{=} x \oplus (\ominus x \oplus y) \\
 &\stackrel{(2)}{=} \text{Exp}_x(\text{PT}_{e \rightarrow x}(\text{Log}_e(\ominus x \oplus y))) \\
 &\stackrel{(3)}{\Rightarrow} \text{Log}_x(y) = \text{PT}_{e \rightarrow x}(\text{Log}_e(\ominus x \oplus y)).
 \end{aligned} \tag{B.32}$$

The above comes from the following.

- (1) Left cancellation law.
- (2) Definition of gyroaddition.
- (3) Applying $\text{Log}_x(\cdot)$ to both sides.

By the last equation, we have

$$\begin{aligned}
 d(x, y) &= \|\text{Log}_x(y)\|_x \\
 &= \|\text{PT}_{e \rightarrow x}(\text{Log}_e(\ominus x \oplus y))\|_x \\
 &\stackrel{(1)}{=} \|\text{Log}_e(\ominus x \oplus y)\|_e \\
 &= d_{\text{gyr}}(x, y),
 \end{aligned} \tag{B.33}$$

where (1) comes from

- Parallel transport preserving the norm [69, Sec. 3.1].
- $\text{PT}_{x \rightarrow e} \circ \text{PT}_{e \rightarrow x}(v) = v, \forall v \in T_e \mathcal{M}$.

□

B.3.4 Proof of Thm. 78

Proof. We denote the Riemannian logarithm, gyration, and gyronorm on $\{\mathcal{M}, g\}$ by Log , gyr , and $\|\cdot\|_{\text{gyr}}$, respectively, while $\widetilde{\text{Log}}$, $\widetilde{\text{gyr}}$, and $\|\cdot\|_{\widetilde{\text{gyr}}}$ are their counterparts on $\{\widetilde{\mathcal{M}}, \widetilde{g}\}$. We recall the following from Nguyen and Yang [159, Lems. 2.1–2.3]:

$$x \oplus y = \phi^{-1}(\phi(x) \widetilde{\oplus} \phi(y)), \tag{B.34}$$

$$t \odot x = \phi^{-1}(t \widetilde{\odot} \phi(x)), \tag{B.35}$$

$$\text{gyr}[x, y]z = \phi^{-1}(\widetilde{\text{gyr}}[\phi(x), \phi(y)]\phi(z)), \tag{B.36}$$

where $x, y, z \in \mathcal{M}$.

Gyrodistance.

$$\begin{aligned}
 \widetilde{d}_{\text{gyr}}(\phi(x), \phi(y)) &= \|\widetilde{\Theta}\phi(x) \oplus \phi(y)\|_{\widetilde{\text{gyr}}} \\
 &\stackrel{(1)}{=} \|\phi(\Theta x \oplus y)\|_{\widetilde{\text{gyr}}} \\
 &= \sqrt{\left\langle \widetilde{\text{Log}}_{\widetilde{e}}(\phi(\Theta x \oplus y)), \widetilde{\text{Log}}_{\widetilde{e}}(\phi(\Theta x \oplus y)) \right\rangle_{\widetilde{e}}} \\
 &\stackrel{(2)}{=} \sqrt{\langle \text{Log}_e(\Theta x \oplus y), \text{Log}_e(\Theta x \oplus y) \rangle_e} \\
 &= \|\Theta x \oplus y\|_{\text{gyr}} \\
 &= d_{\text{gyr}}(x, y).
 \end{aligned} \tag{B.37}$$

The derivation above comes from the following.

(1) By Eqs. (B.34) and (B.35):

$$\phi(\Theta x \oplus y) = \phi(\Theta x) \widetilde{\Theta} \phi(y). \tag{B.38}$$

(2) By the isometry:

$$\text{Log}_x(y) = (\phi_{*,x})^{-1} \left(\widetilde{\text{Log}}_{\phi(x)}(\phi(y)) \right), \forall x, y \in \mathcal{M}, \tag{B.39}$$

$$\langle v, w \rangle_x = \langle \phi_{*,x}(v), \phi_{*,x}(w) \rangle_{\phi(x)}, \forall x \in \mathcal{M} \text{ and } \forall v, w \in T_x \mathcal{M}, \tag{B.40}$$

where $\phi_{*,x}$ is the differential map. Here, the RHSs contain the operators over $\widetilde{\mathcal{M}}$, whereas the LHSs involve those over $\mathcal{M}$.

Gyroiometry. Given any $x, y, z, a \in \mathcal{M}$, we have the following by the isometry of ϕ .

For the gyroinverse:

$$\begin{aligned}
 d_{\widetilde{\text{gyr}}}(\widetilde{\Theta}\phi(x), \widetilde{\Theta}\phi(y)) &= d_{\widetilde{\text{gyr}}}(\phi(\Theta x), \phi(\Theta y)) \\
 &= d_{\text{gyr}}(\Theta x, \Theta y) \\
 &= d_{\text{gyr}}(x, y) \\
 &= d_{\widetilde{\text{gyr}}}(\phi(x), \phi(y)).
 \end{aligned} \tag{B.41}$$

For the gyration:

$$\begin{aligned}
 & d_{\widetilde{\text{gyr}}}(\widetilde{\text{gyr}}[\phi(z), \phi(a)]\phi(x), \widetilde{\text{gyr}}[\phi(z), \phi(a)]\phi(y)) \\
 &= d_{\widetilde{\text{gyr}}}(\phi(\text{gyr}[z, a]x), \phi(\text{gyr}[z, a]y)) \\
 &= d_{\text{gyr}}(\text{gyr}[z, a]x, \text{gyr}[z, a]y) \\
 &= d_{\text{gyr}}(x, y) \\
 &= d_{\widetilde{\text{gyr}}}(\phi(x), \phi(y)).
 \end{aligned} \tag{B.42}$$

For the left gyrotranslation:

$$\begin{aligned}
 & d_{\widetilde{\text{gyr}}}(\phi(z) \widetilde{\oplus} \phi(x), \phi(z) \widetilde{\oplus} \phi(y)) \\
 &= d_{\widetilde{\text{gyr}}}(\phi(z \oplus x), \phi(z \oplus y)) \\
 &= d_{\text{gyr}}(z \oplus x, z \oplus y) \\
 &= d_{\text{gyr}}(x, y) \\
 &= d_{\widetilde{\text{gyr}}}(\phi(x), \phi(y)).
 \end{aligned} \tag{B.43}$$

□

B.3.5 Proof of Thm. 79

We first prove a useful lemma.

Lemma 190 (Left gyrotranslation law). *Every pseudo-reductive gyrogroup $\{G, \oplus\}$ verifies the left gyrotranslation law:*

$$\ominus(x \oplus y) \oplus (x \oplus z) = \text{gyr}[x, y](\ominus y \oplus z), \quad \forall x, y, z \in G. \tag{B.44}$$

Proof. This lemma generalizes Nguyen and Yang [159, Lems. I.1 and L.1], which prove the left gyrotranslation law on the specific gyrogroups of the SPD and Grassmannian manifolds. Their proof only relies on the left cancellation and the basic axioms (G1–G3). Note that the original proof of left gyrotranslation on the Grassmannian [159, Lem. I.1] is questionable, as it relies on the left cancellation of gyrogroups, and the Grassmannian is not a gyrogroup but a non-reductive gyrogroup. Fortunately, as we show in Thm. 75, the general pseudo-reductive gyrogroups, including the Grassmannian, enjoy left cancellation. Therefore, the proof in Nguyen and Yang [159, Lem. I.1] can be readily generalized to pseudo-reductive gyrogroups. □

Proof of Thm. 79. $\Rightarrow$: For any $z, a \in G$, the gyroautomorphism can be expressed by

the gyrator identity in Thm. 75:

$$\text{gyr}[x, y]z = X \oplus \bar{z}, \quad (\text{B.45})$$

$$\text{gyr}[x, y]a = X \oplus \bar{a}, \quad (\text{B.46})$$

where $X = \ominus(x \oplus y)$, $\bar{z} = x \oplus (y \oplus z)$, and $\bar{a} = x \oplus (y \oplus a)$. Then Nguyen and Yang [159, Eq. (30)], derived for the specific Grassmannian, can be directly extended to pseudo-reductive gyrogroups, as it only relies on left gyrotranslation, invariance of the norm under gyroautomorphisms, and the axioms (G1–G3).

$\Leftarrow$:

$$\begin{aligned} \|\text{gyr}[x, y](z)\|_{\text{gyr}} &= \|\text{gyr}[x, y](\ominus e \oplus z)\|_{\text{gyr}} \quad (\text{Case (7) in Thm. 75 indicates } \ominus e = e) \\ &= \|\ominus \text{gyr}[x, y](e) \oplus \text{gyr}[x, y](z)\|_{\text{gyr}} \quad (\text{automorphism}) \\ &= d(\text{gyr}[x, y](e), \text{gyr}[x, y](z)) \\ &= d(e, z) \\ &= \|\ominus e \oplus z\|_{\text{gyr}} \\ &= \|z\|_{\text{gyr}}. \end{aligned} \quad (\text{B.47})$$

□

B.3.6 Proof of Thm. 80

Proof. Given any $x, y, z \in G$, we prove the two claims as follows.

Gyroiometry of the left gyrotranslation. This property generalizes Nguyen and Yang [159, Thms. 2.12 and 2.16], which deal with the gyrotranslation in the SPD and Grassmannian, respectively. We have the following:

$$\begin{aligned} d(L_x(y), L_x(z)) &= d(x \oplus y, x \oplus z) \\ &= \|\ominus(x \oplus y) \oplus (x \oplus z)\|_{\text{gyr}} \\ &= \|\text{gyr}[x, y](\ominus y \oplus z)\|_{\text{gyr}} \quad (\text{left gyrotranslation law}) \\ &= \|\ominus y \oplus z\|_{\text{gyr}} \quad (\text{gyroiometry of the automorphism}) \\ &= d(y, z). \end{aligned} \quad (\text{B.48})$$

Gyroiometry of the gyroinverse.

$$\begin{aligned}
& d(\ominus x, \ominus y) \\
&= \|x \ominus y\|_{\text{gyr}} \\
&= \|\ominus y \oplus x\|_{\text{gyr}} \quad (\text{gyrocommutativity and gyroisometry of the automorphism}) \\
&= d(y, x) \\
&= d(x, y) \quad (\text{symmetry of the geodesic distance}).
\end{aligned}
\tag{B.49}$$

□

B.3.7 Proof of Thm. 81

We give a concise argument based on Thms. 77, 79 and 80.

Proof. First, all these gyrospaces are characterized by Eqs. (2.77) and (2.78). By Thm. 77, their gyrodistances agree with the geodesic distances. It thus remains to establish the gyroisometries.

As shown by Thms. 79 and 80, it suffices to show that gyrations in each space preserve the gyronorm. This argument on the SPD and ONB Grassmannian has already been proven [159, Lems. L.2 and I.2]. Since the ONB Grassmannian is isometric to the PP Grassmannian via Eq. (2.138), Thm. 78 implies that the same arguments apply to the PP. We therefore only need to treat $\mathfrak{st}_K^n$ with $K \leq 0$. As the Euclidean case $\mathbb{R}^n$ is trivial, we only need to show the Poincaré ball. In the following, a, b, x, y are arbitrary points in $\mathbb{P}_K^n$.

Norm invariance under gyrations. As the Poincaré ball forms a real inner-product gyrovector space [200, Def. 6.2 and Thm. 6.85], any gyration preserves the Euclidean norm:

$$\|\text{gyr}[a, b](x)\| = \|x\|, \quad \forall x \in \mathfrak{st}_K^n. \tag{B.50}$$

For the gyronorm, we further have

$$\begin{aligned}
\|\text{gyr}[a, b]x\|_{\text{gyr}} &= \|\text{Log}_{\mathbf{o}}(\text{gyr}[a, b]x)\|_{\mathbf{o}} \\
&= \frac{2}{\sqrt{|K|}} \tanh^{-1} \left(\sqrt{|K|} \|\text{gyr}[a, b]x\| \right) \\
&= \frac{2}{\sqrt{|K|}} \tanh^{-1} \left(\sqrt{|K|} \|x\| \right) \\
&= \|x\|_{\text{gyr}}.
\end{aligned}
\tag{B.51}$$

□

B.3.8 Proof of Thm. 83

Proof. According to Thm. 80, any left gyrotranslation is a gyroisometry. For any $y \in \mathcal{M}$, we have the following:

$$\begin{aligned} d(\beta \oplus x_i, y) &\stackrel{(1)}{=} d(\ominus\beta \oplus (\beta \oplus x_i), \ominus\beta \oplus y) \\ &\stackrel{(2)}{=} d((\ominus\beta \oplus \beta) \oplus \text{gyr}[\ominus\beta, \beta](x_i), \ominus\beta \oplus y) \\ &\stackrel{(3)}{=} d(x_i, \ominus\beta \oplus y). \end{aligned} \tag{B.52}$$

The above comes from the following.

- (1) Any left gyrotranslation is a gyroisometry.
- (2) Left gyroassociative law.
- (3) $\ominus\beta \oplus \beta = e$ and pseudo-reduction.

Denoting the gyromean of $\{x_i\}$ and $\{\beta \oplus x_i\}$ as μ and $\tilde{\mu}$, we have the following:

$$\begin{aligned} \beta \oplus \mu &\stackrel{(1)}{=} \beta \oplus (\ominus\beta \oplus \tilde{\mu}) \\ &\stackrel{(2)}{=} \text{gyr}[\beta, \ominus\beta](\tilde{\mu}) \\ &\stackrel{(3)}{=} \tilde{\mu}. \end{aligned} \tag{B.53}$$

The above comes from the following.

- (1) Eq. (B.52) indicates that $\mu = \ominus\beta \oplus \tilde{\mu}$.
- (2) Left gyroassociative law.
- (3) Pseudo-reduction.

Algorithm 4: ONB Grassmannian logarithm [19, Alg. 5.3]

Input: $U, Y \in \text{Gr}(p, n)$ are Stiefel representatives under the ONB perspective.
 $QSR^\top \stackrel{\text{SVD}}{:=} Y^\top U$ with S in ascending order, and Q and R flipped column-wise
 accordingly $\hat{S} = \sqrt{I_p - S^2}$ $\Delta = (I_n - UU^\top)YQ^{\frac{\arcsin(\hat{S})}{\hat{S}}}R^\top$

Output: $\text{Log}_U(Y) = \Delta$

Now, we proceed to deal with the second property. We have the following:

$$\begin{aligned}
 d(t \odot x_i, e) &\stackrel{(1)}{=} \|\ominus e \oplus (t \odot x_i)\|_{\text{gyr}} \\
 &\stackrel{(2)}{=} \|t \odot x_i\|_{\text{gyr}} \\
 &= \|t \text{Log}_e(x_i)\|_e \\
 &= |t| \|\text{Log}_e(x_i)\|_e \\
 &= |t| \|x_i\|_{\text{gyr}} \\
 &\stackrel{(3)}{=} |t| \|\ominus e \oplus x_i\|_{\text{gyr}} \\
 &= |t| d(e, x_i) \\
 &\stackrel{(4)}{=} |t| d(x_i, e)
 \end{aligned} \tag{B.54}$$

The above follows from the following.

- (1) Symmetry of gyrodistance (as geodesic distance).
- (2) $\ominus e = e$.
- (3) $x_i = \ominus e \oplus x_i$.
- (4) Symmetry of gyrodistance (as geodesic distance).

The last equation in Eq. (B.54) indicates the homogeneity of dispersion from e . $\square$

B.3.9 Proof of Thm. 86

We first review a fast and stable algorithm for the ONB Grassmannian logarithm [19, Alg. 5.3], and the calculation of Grassmannian logarithm under the projector perspective by the ONB Grassmannian logarithm [161, Prop. 3.12].

Alg. 4 reviews a fast and stable algorithm for the Grassmannian Riemannian logarithm under the ONB perspective $\text{Gr}(p, n)$. The vanilla Riemannian logarithm in Tabs. 2.10 and 2.11 requires an $n \times p$ SVD and a $p \times p$ matrix inverse, while Alg. 4 only requires a $p \times p$ SVD. Therefore, Alg. 4 is more efficient than the vanilla logarithm.

Besides, Alg. 4 can also return a unique tangent vector when Y is in the cut locus of U . For more details, please refer to Bendokat et al. [19, Sec. 5.2].

As the projector perspective is isometric to the ONB perspective, the Grassmannian logarithm under the projector perspective can be calculated by the ONB Grassmannian logarithm [161, Prop. 3.12].

Proposition 191 ([161]). *Given any $P, Q \in \widetilde{\text{Gr}}(p, n)$ with $U = \pi^{-1}(P)$ and $V = \pi^{-1}(Q)$, the Riemannian logarithm $\widetilde{\text{Log}}_P(Q)$ on $\widetilde{\text{Gr}}(p, n)$ is given as*

$$\widetilde{\text{Log}}_P(Q) = \pi_{*,U}(\text{Log}_U V), \quad (\text{B.55})$$

where Log is the Riemannian logarithm under the ONB perspective, $\pi_{*,U} : T_U \text{Gr}(p, n) \rightarrow T_P \widetilde{\text{Gr}}(p, n)$ is the differential map of π at U , which is defined as

$$\pi_{*,U}(\Delta) = \Delta U^\top + U \Delta^\top, \forall \Delta \in T_U \text{Gr}(p, n). \quad (\text{B.56})$$

Now, we begin to present the proof.

Proof of Thm. 86. We first show the expression for $\text{Log}_{I_{p,n}}$ and $\widetilde{\text{Log}}_{\tilde{I}_{p,n}}$.

First note the following:

$$(I_n - I_{p,n} I_{p,n}^\top) = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{n-p} \end{pmatrix}, \quad (\text{B.57})$$

$$\begin{aligned} U^\top I_{p,n} &= (U_1^\top, U_2^\top) \begin{pmatrix} I_p \\ \mathbf{0} \end{pmatrix} \\ &= U_1^\top, \end{aligned} \quad (\text{B.58})$$

By the above two equations, the ONB Grassmannian logarithm at $I_{p,n}$ is

$$\begin{aligned} \text{Log}_{I_{p,n}}(U) &= \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{n-p} \end{pmatrix} \begin{pmatrix} U_1 \\ U_2 \end{pmatrix} Q \frac{\arcsin(\hat{S})}{\hat{S}} R^\top \quad (\text{Alg. 4}) \\ &= \begin{pmatrix} \mathbf{0} \\ U_2 Q \frac{\arcsin(\hat{S})}{\hat{S}} R^\top \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{0} \\ \tilde{U}_2 \end{pmatrix}, \end{aligned} \quad (\text{B.59})$$

where $QSR^\top \stackrel{\text{SVD}}{:=} U_1^\top$ with S in ascending order, and Q and R flipped column-wise

accordingly, and $\hat{S} = \sqrt{I_p - S^2}$.

For $\widetilde{\text{Log}}_{\tilde{I}_{p,n}}$, we have

$$\begin{aligned} \widetilde{\text{Log}}_{\tilde{I}_{p,n}}(UU^\top) &\stackrel{(1)}{=} \pi_{*, I_{p,n}} \left(\text{Log}_{I_{p,n}}(U) \right) \\ &\stackrel{(2)}{=} \pi_{*, I_{p,n}} \left(\begin{pmatrix} \mathbf{0} \\ \tilde{U}_2 \end{pmatrix} \right) \\ &\stackrel{(3)}{=} \begin{pmatrix} \mathbf{0} & \tilde{U}_2^\top \\ \tilde{U}_2 & \mathbf{0} \end{pmatrix}. \end{aligned} \tag{B.60}$$

The above derivation comes from the following.

(1) Thm. 191

(2) Eq. (B.59)

(3) For any $\Delta = (\Delta_1^\top, \Delta_2^\top)^\top \in T_{I_{p,n}} \text{Gr}(p, n)$, where Δ_1 is $p \times p$, we have the following:

$$\begin{aligned} \pi_{*, I_{p,n}} \left(\begin{pmatrix} \Delta_1 \\ \Delta_2 \end{pmatrix} \right) &= \begin{pmatrix} \Delta_1 \\ \Delta_2 \end{pmatrix} (I_p, \mathbf{0}) + \begin{pmatrix} I_p \\ \mathbf{0} \end{pmatrix} (\Delta_1^\top, \Delta_2^\top) \\ &= \begin{pmatrix} \Delta_1 & \mathbf{0} \\ \Delta_2 & \mathbf{0} \end{pmatrix} + \begin{pmatrix} \Delta_1^\top & \Delta_2^\top \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \\ &= \begin{pmatrix} \Delta_1 + \Delta_1^\top & \Delta_2^\top \\ \Delta_2 & \mathbf{0} \end{pmatrix}. \end{aligned} \tag{B.61}$$

Combining all the above results together, we have the following:

$$\begin{aligned} [\overline{UU^\top}, \tilde{I}_{p,n}] &= [\widetilde{\text{Log}}_{\tilde{I}_{p,n}}(UU^\top), \tilde{I}_{p,n}] \\ &= \left[\begin{pmatrix} \mathbf{0} & \tilde{U}_2^\top \\ \tilde{U}_2 & \mathbf{0} \end{pmatrix}, \tilde{I}_{p,n} \right] \\ &= \begin{pmatrix} \mathbf{0} & \tilde{U}_2^\top \\ \tilde{U}_2 & \mathbf{0} \end{pmatrix} \begin{pmatrix} I_p & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} - \begin{pmatrix} I_p & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{0} & \tilde{U}_2^\top \\ \tilde{U}_2 & \mathbf{0} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{0} & -\tilde{U}_2^\top \\ \tilde{U}_2 & \mathbf{0} \end{pmatrix}. \end{aligned} \tag{B.62}$$

□

B.3.10 Proof of Thm. 88

Proof. Given $v \in T_{\mathbf{0}}\mathfrak{st}_K^n$, we have the following:

$$\lambda_{\mathbf{0}}^K = 2, \quad (\text{B.63})$$

$$\text{Log}_{\mathbf{0}}(y) = \tan_K^{-1} \left(\sqrt{|K|} \|y\| \right) \frac{y}{\sqrt{|K|} \|y\|}, \quad (\text{B.64})$$

$$\text{PT}_{\mathbf{0} \rightarrow x}(v) = \frac{\lambda_{\mathbf{0}}^K}{\lambda_x^K} \text{gyr}[x, -\mathbf{0}](v) = \frac{2}{\lambda_x^K} v, \quad (\text{B.65})$$

$$\text{Exp}_{\mathbf{0}}(v) = \tan_K \left(\sqrt{|K|} \|v\| \right) \frac{v}{\sqrt{|K|} \|v\|}. \quad (\text{B.66})$$

Therefore, we have

$$\begin{aligned} \text{Exp}_x(\text{PT}_{\mathbf{0} \rightarrow x}(\text{Log}_{\mathbf{0}}(y))) &= \text{Exp}_x \left(\frac{2}{\sqrt{|K|} \lambda_x^K} \tan_K^{-1} \left(\sqrt{|K|} \|y\| \right) \frac{y}{\|y\|} \right) \\ &= x \oplus_K y, \end{aligned} \quad (\text{B.67})$$

which implies

$$\text{Exp}_{\mathbf{0}}(t \text{Log}_{\mathbf{0}}(x)) = t \odot_K x. \quad (\text{B.68})$$

□

B.3.11 Proof of Thm. 89

Proof. We only need to show the case of $K > 0$. Let x, y, z, w be any vectors in $\mathfrak{st}_K^n$, and $s, t \in \mathbb{R}$ be real scalars.

We first notice the gyroaddition is a linear combination:

$$x \oplus_K y = \frac{(1 - 2K \langle x, y \rangle - K \|y\|^2)x + (1 + K \|x\|^2)y}{1 - 2K \langle x, y \rangle + K^2 \|x\|^2 \|y\|^2} = \frac{Ax + By}{D}, \quad (\text{B.69})$$

where

$$\begin{aligned} A &= 1 - 2K \langle x, y \rangle - K \|y\|^2, \\ B &= 1 + K \|x\|^2, \\ D &= 1 - 2K \langle x, y \rangle + K^2 \|x\|^2 \|y\|^2. \end{aligned} \quad (\text{B.70})$$

Recalling Eq. (2.146), the gyration $\text{gyr}[x, y](z)$ is also a linear expression:

$$\text{gyr}[x, y](z) = z + f_1 \cdot x + f_2 \cdot y. \quad (\text{B.71})$$

Axioms G1–G3. Since $e = \mathbf{0}$ and $\ominus_K x = -x$, G1 and G2 can be immediately verified. The left gyroassociative law follows from the definition of gyration [13, Eq. 36]:

$$\text{gyr}[x, y](z) = -(x \oplus_K y) \oplus_K (x \oplus_K (y \oplus_K z)). \quad (\text{B.72})$$

To confirm that any gyration is an automorphism of $(\mathfrak{st}_K^n, \oplus_K)$, we verify the following identity using symbolic computation:

$$\text{gyr}[x, y](w \oplus_K z) = \text{gyr}[x, y](w) \oplus_K \text{gyr}[x, y](z). \quad (\text{B.73})$$

Expanding all necessary inner products (e.g., $\langle x, w \oplus_K z \rangle$, $\langle y, w \oplus_K z \rangle$) and norms in terms of $\langle x, w \rangle$, $\langle x, z \rangle$, etc., we express both sides of Eq. (B.73) as linear combinations:

$$\text{gyr}[x, y](w \oplus_K z) = f_1 x + f_2 y + f_3 w + f_4 z, \quad (\text{B.74})$$

$$\text{gyr}[x, y](w) \oplus_K \text{gyr}[x, y](z) = f'_1 x + f'_2 y + f'_3 w + f'_4 z. \quad (\text{B.75})$$

We use SymPy to compare the coefficients of x , y , w , and z on both sides. The above is exposed in `stereographic_gyr_automorphism.py`.

(G4) Left reduction law. Similarly, we expand $\text{gyr}[x \oplus_K z, y](w)$ and $\text{gyr}[x, y](w)$ and compare the coefficients, as implemented in `stereographic_left_reduction.py`.

Gyrocommutative law. This has been verified by Bachmann et al. [13, Lem. 11].

(V1) Identity scalar multiplication. This can be directly verified by definition:

$$t \odot_K x = \frac{\tan\left(t \tan^{-1}(\sqrt{K} \|x\|)\right)}{\sqrt{K}} \frac{x}{\|x\|}, \forall x \neq 0. \quad (\text{B.76})$$

(V2) Scalar distributive law. We want to verify

$$(s + t) \odot_K x = (s \odot_K x) \oplus_K (t \odot_K x). \quad (\text{B.77})$$

We only need to show the case of $x \neq \mathbf{0}$. Let $\theta = \tan^{-1}(\sqrt{K} \|x\|)$. Then scalar multiplication is simplified as

$$t \odot_K x = \frac{\tan(t\theta)}{\sqrt{K} \|x\|} x. \quad (\text{B.78})$$

Using symbolic computation (see `stereographic_gyr_v2.py`), we obtain the following

w.r.t. Eq. (B.77):

$$\text{LHS} = \frac{\tan((s+t)\theta)}{\sqrt{K} \|x\|}, \quad (\text{B.79})$$

$$\text{RHS} = \frac{\tan(s\theta) + \tan(t\theta)}{\sqrt{K} \|x\| (1 - \tan(s\theta) \tan(t\theta))}. \quad (\text{B.80})$$

The identity follows from the tangent addition formula:

$$\tan((s+t)\theta) = \frac{\tan(s\theta) + \tan(t\theta)}{1 - \tan(s\theta) \tan(t\theta)}. \quad (\text{B.81})$$

(V3) Scalar associative law. This can be directly verified by definition.

(V4) Gyroautomorphism. We now verify

$$\text{gyr}[x, y](t \odot_K z) = t \odot_K \text{gyr}[x, y](z). \quad (\text{B.82})$$

Let us denote

$$\alpha_t^{(z)} = \frac{\tan\left(t \tan^{-1}\left(\sqrt{K} \|z\|\right)\right)}{\sqrt{K} \|z\|}. \quad (\text{B.83})$$

By the linearity of gyration [13, Lem. 11], the left-hand side of Eq. (B.82) becomes

$$\text{LHS} = \alpha_t^{(z)} \text{gyr}[x, y](z), \quad (\text{B.84})$$

while the right-hand side reads

$$\text{RHS} = \alpha_t^{\text{gyr}[x, y](z)} \text{gyr}[x, y](z). \quad (\text{B.85})$$

Since $\alpha_t^{(\cdot)}$ depends only on the norm of its argument, it suffices to show

$$\|z\| = \|\text{gyr}[x, y](z)\|, \quad (\text{B.86})$$

which holds as proven by Bachmann et al. [13, Lem. 11, iv)].

(V5) Identity gyroautomorphism. In each of the three special cases when (i) $x = \mathbf{0}$, or (ii) $y = \mathbf{0}$, or (iii) x and y are parallel in $\mathbb{V}$, $x \parallel y$, we have

$$\text{gyr}[\mathbf{0}, x](z) \stackrel{(1)}{=} z, \quad (\text{B.87})$$

$$\text{gyr}[x, \mathbf{0}](z) \stackrel{(2)}{=} z, \quad (\text{B.88})$$

$$\text{gyr}[x, y](z) \stackrel{(3)}{=} z, \quad x \parallel y, \quad (\text{B.89})$$

where (1–3) come from $A_{\text{st}}x + B_{\text{st}}y = \mathbf{0}$ in Eq. (2.146). Therefore, we have

$$\text{gyr}[s \odot_K x, t \odot_K x] = \text{gyr}[\alpha_s^{(x)}x, \alpha_t^{(x)}x] = \text{id}. \quad (\text{B.90})$$

Note that (1–2) are also implied by the first gyrogroup theorem [200, Thm. 2.10]. $\square$

B.3.12 Proof of Thm. 90

This proof largely follows the one for Thm. 81.

Proof. Norm invariance under gyrations. By Bachmann et al. [13, Lem. 11], any gyration preserves the Euclidean norm:

$$\|\text{gyr}[a, b](x)\| = \|x\|, \quad \forall x \in \mathfrak{st}_K^n. \quad (\text{B.91})$$

For the gyronorm, we further have

$$\begin{aligned} \|\text{gyr}[a, b]x\|_{\text{gyr}} &= \|\text{Log}_{\mathbf{0}}(\text{gyr}[a, b]x)\|_{\mathbf{0}} \\ &= \frac{2}{\sqrt{|K|}} \tan_K^{-1} \left(\sqrt{|K|} \|\text{gyr}[a, b]x\| \right) \\ &= \frac{2}{\sqrt{|K|}} \tan_K^{-1} \left(\sqrt{|K|} \|x\| \right) \\ &= \|x\|_{\text{gyr}}, \end{aligned} \quad (\text{B.92})$$

with $\tan_K = \tanh$ for $K < 0$ and $\tan_K = \tan$ for $K > 0$.

Isometry of left gyrotranslation and gyroinverse. As shown in Thm. 89, $\mathfrak{st}_K^n$ forms a gyrocommutative gyrogroup. By Thm. 80, the left gyrotranslation and gyroinverse are gyroisometries. $\square$

B.3.13 Proof of Thm. 91

Proof. Non-singular cases. Denoting $\|x\|_s = \|x_s\|$, $\forall x \in \mathcal{M}_K^n$, we have

$$(t \text{Log}_{\mathbf{0}} x)_s = t \frac{\cos_K^{-1}(\sqrt{|K|}x_t)}{\sqrt{|K|} \|x_s\|} x_s, \quad (\text{B.93})$$

$$\|t \text{Log}_{\mathbf{0}} x\|_s = |t| \frac{\cos_K^{-1}(\sqrt{|K|}x_t)}{\sqrt{|K|}}, \quad (\text{B.94})$$

$$\sqrt{|K|} \|t \operatorname{Log}_{\bar{\mathbf{0}}} x\|_s = |t| \cos_K^{-1}(\sqrt{|K|} x_t), \quad (\text{B.95})$$

For $t \neq 0$, the normalized spatial direction satisfies

$$\frac{(t \operatorname{Log}_{\bar{\mathbf{0}}} x)_s}{\|t \operatorname{Log}_{\bar{\mathbf{0}}} x\|_s} = \operatorname{sgn}(t) \frac{x_s}{\|x_s\|}. \quad (\text{B.96})$$

Since $\cos_K$ is even and $\sin_K$ is odd, substituting the above expressions into $\operatorname{Exp}_{\bar{\mathbf{0}}}$ in Tab. 2.14 recovers the signed argument $t \cos_K^{-1}(\sqrt{|K|} x_t)$ in Eq. (3.55). The cases $t = 0$ and $x = \bar{\mathbf{0}}$ are handled by the first branch of Eq. (3.55).

For $t = -1$, we have

$$\begin{aligned} -1 \odot_K^{\mathcal{M}} x &= \frac{1}{\sqrt{|K|}} \left[\frac{\cos_K \left(-\cos_K^{-1}(\sqrt{|K|} x_t) \right)}{\frac{\sin_K \left(-\cos_K^{-1}(\sqrt{|K|} x_t) \right)}{\|x_s\|}} x_s \right] \\ &\stackrel{(1)}{=} \left[\frac{x_t}{-\frac{\sin_K \left(\cos_K^{-1}(\sqrt{|K|} x_t) \right)}{\sqrt{|K|} \|x_s\|}} x_s \right], \end{aligned} \quad (\text{B.97})$$

where (1) comes from $\cos_K(-\theta) = \cos_K(\theta)$ and $\sin_K(-\theta) = -\sin_K(\theta)$. The rest is to show $\frac{\sin_K \left(\cos_K^{-1}(\sqrt{|K|} x_t) \right)}{\sqrt{|K|} \|x_s\|} = 1$:

$$\frac{\sin_K \left(\cos_K^{-1}(\sqrt{|K|} x_t) \right)}{\sqrt{|K|} \|x_s\|} \stackrel{(1)}{=} \frac{\sqrt{\operatorname{sign}(K)(1 - |K| x_t^2)}}{\sqrt{|K|} \|x_s\|} \stackrel{(2)}{=} 1. \quad (\text{B.98})$$

The derivation above is based on the following:

(1) $\cos_K^2(\theta) + \operatorname{sign}(K) \sin_K^2(\theta) = 1$ implies

$$\sin_K(\theta) = \sqrt{\operatorname{sign}(K)(1 - \cos_K^2(\theta))}, \forall \theta > 0. \quad (\text{B.99})$$

Considering $\cos_K^{-1}(\theta) \geq 0$ for any $\theta \in \operatorname{dom}(\cos_K^{-1}(\cdot))$, we have (1).

(2)

$$\begin{aligned} \langle x, x \rangle_K &= \operatorname{sign}(K) x_t^2 + \|x_s\|^2 = \frac{1}{K} \\ \Rightarrow K \|x_s\|^2 &= 1 - |K| x_t^2 \\ \Rightarrow |K| \|x_s\|^2 &= \operatorname{sign}(K)(1 - |K| x_t^2). \end{aligned} \quad (\text{B.100})$$

Equivalence in the singular cases. By Eq. (3.50),

$$\sqrt{K} \|u\| = \frac{\sqrt{K} \|x_s\|}{1 + \sqrt{K} x_t}. \quad (\text{B.101})$$

Writing $\theta = \cos^{-1}(\sqrt{K} x_t)$ or $\sqrt{K} x_t = \cos \theta$, the sphere constraint gives $\sqrt{K} \|x_s\| = \sin \theta$. Hence

$$\sqrt{K} \|u\| = \frac{\sin \theta}{1 + \cos \theta} = \tan \frac{\theta}{2}, \quad \Rightarrow \tan^{-1}(\sqrt{K} \|u\|) = \frac{\theta}{2}. \quad (\text{B.102})$$

Therefore, (i) is equivalent to (ii) since $t \tan^{-1}(\sqrt{K} \|u\|) = \frac{t\theta}{2}$.

Next, we use the closed form of gyromultiplication Eq. (3.58). For $K > 0$, the gyromultiplication reads

$$t \odot_K^{\mathcal{M}} x = \frac{1}{\sqrt{K}} \begin{bmatrix} \cos(t\theta) \\ \frac{\sin(t\theta)}{\|x_s\|} x_s \end{bmatrix}. \quad (\text{B.103})$$

If (ii) holds, then $\sin(t\theta) = 0$ and $\cos(t\theta) = -1$, whence $t \odot_K^{\mathcal{M}} x = [-1/\sqrt{K}, 0]^\top = -\bar{0}$. $\square$

B.3.14 Proof of Thm. 92

Proof. We denote $x \oplus_K^{\mathcal{M}} y = [z_t, z_s^\top]^\top$. As the results are trivial under $x = \bar{0}$ or $y = \bar{0}$, we assume $x \neq \bar{0}$ and $y \neq \bar{0}$ in the following.

Non-singular cases. We first consider the non-singular case: i) $K < 0$; ii) $K > 0$, $x, y \neq \pm \bar{0}$, and $u \neq \frac{v}{K\|v\|^2}$. Since gyroadditions on $\mathcal{M}_K^n$ and $\mathfrak{st}_K^n$ are both defined by Eq. (2.77), we have the following under isometries [159, Lem. 2.2]:

$$x \oplus_K^{\mathcal{M}} y = \pi_{\mathfrak{st}_K^n \rightarrow \mathcal{M}_K^n} \left(\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(x) \oplus_K \pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(y) \right). \quad (\text{B.104})$$

Following Eq. (B.70), we rewrite the gyroaddition on the stereographic model as

$$u \oplus_K v = \frac{(1 - 2K\langle u, v \rangle - K\|v\|^2)u + (1 + K\|u\|^2)v}{1 - 2K\langle u, v \rangle + K^2\|u\|^2\|v\|^2} = \frac{Au + Bv}{\Delta}, \quad (\text{B.105})$$

where

$$A = 1 - 2K\langle u, v \rangle - K\|v\|^2, \quad B = 1 + K\|u\|^2, \quad \Delta = 1 - 2K\langle u, v \rangle + K^2\|u\|^2\|v\|^2. \quad (\text{B.106})$$

Using $\langle u, v \rangle = s_{xy}/(ab)$, $\|u\|^2 = n_x/a^2$, $\|v\|^2 = n_y/b^2$, one obtains

$$A = \frac{ab^2 - 2Kbs_{xy} - Kan_y}{ab^2}, \quad B = \frac{a^2 + Kn_x}{a^2}, \quad \Delta = \frac{D}{a^2b^2}. \quad (\text{B.107})$$

Hence

$$\mathbf{st}_K^n \ni w = u \oplus_K v = \frac{Au + Bv}{\Delta} = \frac{1}{\Delta} \left(\frac{A}{a}x_s + \frac{B}{b}y_s \right). \quad (\text{B.108})$$

Apply $\pi_{\mathbf{st}_K^n \rightarrow \mathcal{M}_K^n}$ to w :

$$z_t = \frac{1}{\sqrt{|K|}} \frac{1 - K\|w\|^2}{1 + K\|w\|^2}, \quad z_s = \frac{2w}{1 + K\|w\|^2}. \quad (\text{B.109})$$

A direct expansion (`radius_gyroaddition.py`) yields

$$\|w\|^2 = \frac{A^2\|u\|^2 + B^2\|v\|^2 + 2AB\langle u, v \rangle}{\Delta^2} = \frac{N}{D}. \quad (\text{B.110})$$

Substituting the above into Eq. (B.109) gives

$$z_t = \frac{1}{\sqrt{|K|}} \frac{1 - KN/D}{1 + KN/D} = \frac{1}{\sqrt{|K|}} \frac{D - KN}{D + KN}. \quad (\text{B.111})$$

For z_s , using Eq. (B.108), Eq. (B.107) and $1 + K\|w\|^2 = (D + KN)/D$,

$$\begin{aligned} z_s &= \frac{2}{1 + KN/D} \cdot \frac{1}{\Delta} \left(\frac{A}{a}x_s + \frac{B}{b}y_s \right) \\ &= \frac{2}{D + KN} ((Aab^2)x_s + (Ba^2b)y_s) \\ &= \frac{2(A_sx_s + A_yy_s)}{D + KN}. \end{aligned} \quad (\text{B.112})$$

Gyroaddition in singular cases ($K > 0$). We show that the definition Eq. (3.59) indeed returns $-\bar{\mathbf{0}}$ in this case.

Step 1: $\text{Log}_{\bar{\mathbf{0}}}(y)$. Write $R = 1/\sqrt{K}$ and choose the polar angle $\theta \in (0, \pi)$ so that

$$x = \begin{bmatrix} R \cos \theta \\ R \sin \theta \hat{s} \end{bmatrix}, \quad y = \begin{bmatrix} -R \cos \theta \\ R \sin \theta \hat{s} \end{bmatrix}, \quad \hat{s} = \frac{x_s}{\|x_s\|}. \quad (\text{B.113})$$

Using $\text{Log}_{\bar{\mathbf{0}}}$ in Tab. 2.14,

$$\text{Log}_{\bar{\mathbf{0}}}(y) = \begin{bmatrix} 0 \\ \frac{\cos^{-1}(\sqrt{K}y_t)}{\sqrt{K}\|y_s\|}y_s \end{bmatrix} = \begin{bmatrix} 0 \\ (\pi - \theta)R\hat{s} \end{bmatrix}. \quad (\text{B.114})$$

Step 2: Parallel transport. With $1 + \sqrt{K}x_t = 1 + \cos \theta \neq 0$ (since $x \neq -\bar{\mathbf{0}}$) and $\text{PT}_{\bar{\mathbf{0}} \rightarrow x}$ in Tab. 2.14, we have

$$\text{PT}_{\bar{\mathbf{0}} \rightarrow x}(\text{Log}_{\bar{\mathbf{0}}}(y)) = \begin{bmatrix} 0 \\ (\pi - \theta)R\hat{s} \end{bmatrix} - \frac{K\langle x_s, (\pi - \theta)R\hat{s} \rangle}{1 + \sqrt{K}x_t} \begin{bmatrix} x_t + \frac{1}{\sqrt{K}} \\ x_s \end{bmatrix}. \quad (\text{B.115})$$

Using $\langle x_s, \hat{s} \rangle = \|x_s\| = R \sin \theta$, $KR^2 = 1$, and $x_t = R \cos \theta$, the scalar factor in Eq. (B.115) equals

$$\frac{K(\pi - \theta)R\langle x_s, \hat{s} \rangle}{1 + \sqrt{K}x_t} = \frac{(\pi - \theta) \sin \theta}{1 + \cos \theta}. \quad (\text{B.116})$$

A short simplification then yields the compact form

$$w = \text{PT}_{\bar{\mathbf{0}} \rightarrow x}(\text{Log}_{\bar{\mathbf{0}}}(y)) = R(\pi - \theta) \begin{bmatrix} -\sin \theta \\ \cos \theta \hat{s} \end{bmatrix}. \quad (\text{B.117})$$

Note that $\|w\|_x = R(\pi - \theta)$, so with $\alpha = \sqrt{K}\|w\|_x$ we have $\alpha = \pi - \theta$.

Step 3: Exponential at x . By Tab. 2.14,

$$\text{Exp}_x(w) = \cos(\alpha)x + \frac{\sin(\alpha)}{\alpha}w, \quad \alpha = \sqrt{K}\|w\|_x, \quad (\text{B.118})$$

and here $\cos(\alpha) = \cos(\pi - \theta) = -\cos \theta$, $\sin(\alpha) = \sin(\pi - \theta) = \sin \theta$. Substituting Eq. (B.113) and Eq. (B.117) into Eq. (B.118),

$$\text{Exp}_x(w) = R \left[-\cos \theta \begin{bmatrix} \cos \theta \\ \sin \theta \hat{s} \end{bmatrix} + \sin \theta \begin{bmatrix} -\sin \theta \\ \cos \theta \hat{s} \end{bmatrix} \right] = R \begin{bmatrix} -1 \\ 0 \end{bmatrix} = -\bar{\mathbf{0}}. \quad (\text{B.119})$$

We conclude that $x \oplus_K^{\mathcal{M}} y = -\bar{\mathbf{0}}$ in the singular configuration.

Equivalence in singular cases ($K > 0$). Recalling the isometry between $\mathcal{M}_K^n$ and $\mathfrak{st}_K^n$, we write $u = \frac{x_s}{a}$ and $v = \frac{y_s}{b}$.

(i) $\Rightarrow$ (ii). Write $\alpha = K \|v\|^2 > 0$. From $u = \frac{v}{\alpha}$ and Eq. (3.51) we get

$$\begin{aligned} y_t &= \frac{1}{\sqrt{K}} \frac{1 - \alpha}{1 + \alpha}, \\ x_t &= \frac{1}{\sqrt{K}} \frac{1 - \frac{1}{\alpha}}{1 + \frac{1}{\alpha}} = -\frac{1}{\sqrt{K}} \frac{1 - \alpha}{1 + \alpha} = -y_t, \\ x_s &= \frac{2u}{1 + K\|u\|^2} = \frac{2v/\alpha}{1 + K\|v\|^2/\alpha^2} = \frac{2v}{1 + \alpha} = y_s, \end{aligned} \quad (\text{B.120})$$

Hence $x_s = y_s$ and $x_t = -y_t$.

(ii) $\Rightarrow$ (iii). Under $x_s = y_s$ and $x_t = -y_t$, set $n = \|x_s\|^2 = \|y_s\|^2$ and note $s_{xy} = n$, $n_x = n_y = n$. Then,

$$D = a^2b^2 - 2Kabs_{xy} + K^2n_xn_y = (ab - Kn)^2. \quad (\text{B.121})$$

On the sphere $\mathbb{S}_K^n$, we have the constraint $x_t^2 + \|x_s\|^2 = 1/K$. Since $y_t = -x_t$ and $\|y_s\|^2 = \|x_s\|^2 = n$,

$$\begin{aligned} ab &= (1 + \sqrt{K}x_t)(1 + \sqrt{K}y_t) \\ &= (1 + \sqrt{K}x_t)(1 - \sqrt{K}x_t) \\ &= 1 - Kx_t^2 \\ &= 1 - K \left(\frac{1}{K} - n \right) = Kn. \end{aligned} \quad (\text{B.122})$$

Therefore $D = (ab - Kn)^2 = 0$.

(iii) $\Rightarrow$ (i). This can be obtained using the Cauchy–Schwarz inequality, as in Bachmann et al. [13, App. C.2.1].

Consequences in singular cases ($K > 0$). Under (ii) we also have $a + b = 2$, so

$$N = a^2n + 2abn + b^2n = (a + b)^2n = 4n > 0. \quad (\text{B.123})$$

Using $D = 0$ and $N > 0$, Eq. (B.111) gives

$$z_t = \frac{1}{\sqrt{K}} \frac{-KN}{KN} = -\frac{1}{\sqrt{K}}. \quad (\text{B.124})$$

Besides, since for $x_s = y_s$ one checks $A_s + A_y = (a + b)(ab - Kn) = 0$, Eq. (B.112) yields $z_s = 0$. On the other hand, $u \oplus_K v = \infty$. $\square$

B.3.15 Proof of Thm. 93

Proof. This is already implied by the proofs of Thms. 91 and 92. $\square$

B.3.16 Proof of Thm. 95

Proof. The gyrovector space over the stereographic model is also defined by Eqs. (2.77) and (2.78):

$$\begin{aligned} x \oplus_K y &= \text{Exp}_x (\text{PT}_{\mathbf{0} \rightarrow x} (\text{Log}_{\mathbf{0}}(y))), \\ r \odot_K x &= \text{Exp}_{\mathbf{0}} (r \text{Log}_{\mathbf{0}}(x)), \end{aligned} \quad (\text{B.125})$$

with $x, y \in \mathfrak{st}_K^n$ and $r \in \mathbb{R}$. Since $\pi_{\mathcal{M}_K^n \rightarrow \mathfrak{st}_K^n}(\bar{\mathbf{0}}) = \mathbf{0}$ and $(\mathfrak{st}_K^n, \oplus_K, \odot_K)$ is a gyrovector space, $(\mathcal{M}_K^n, \oplus_K^{\mathcal{M}}, \odot_K^{\mathcal{M}})$ is also a gyrovector space [159, Thm. 2.4]. $\square$

B.3.17 Proof of Thm. 97

Proof. Isometries. The Beltrami–Klein model is isometric to the hyperboloid model by the following diffeomorphisms [131, Thm. 3.7]:

$$\pi_{\mathbb{K}_K^n \rightarrow \mathbb{H}_K^n} : \mathbb{K}_K^n \ni x \mapsto \left(\frac{1}{\sqrt{-K} \sqrt{1 + K\|x\|^2}}, \frac{x}{\sqrt{1 + K\|x\|^2}} \right) \in \mathbb{H}_K^n, \quad (\text{B.126})$$

$$\pi_{\mathbb{H}_K^n \rightarrow \mathbb{K}_K^n} : \mathbb{H}_K^n \ni \begin{bmatrix} x_t \\ x_s \end{bmatrix} \mapsto \frac{x_s}{\sqrt{-K} x_t} \in \mathbb{K}_K^n. \quad (\text{B.127})$$

Combining the isometries Eqs. (B.126), (B.127), (3.50) and (3.51), one can readily obtain the isometries between Beltrami–Klein and Poincaré ball models. Now, we turn to the differential maps. Given a curve over $c(t) \in \mathcal{M}$ with $c(0) = x$ and $c'(0) = v$, the

differential maps can be calculated by

$$\begin{aligned}
 \left. \frac{d\pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n}(c(t))}{dt} \right|_{t=0} &= \left. \frac{d}{dt} \frac{c(t)}{\left(1 + \sqrt{1 + K\|c(t)\|^2}\right)} \right|_{t=0} \\
 &= \frac{\left(1 + \sqrt{1 + K\|x\|^2}\right) v - (1 + K\|x\|^2)^{-\frac{1}{2}} K \langle x, v \rangle x}{\left(1 + \sqrt{1 + K\|x\|^2}\right)^2} \\
 &= \frac{1}{1 + \sqrt{1 + K\|x\|^2}} v - \frac{K \langle x, v \rangle}{\left(1 + \sqrt{1 + K\|x\|^2}\right)^2 \sqrt{1 + K\|x\|^2}} x, \\
 \left. \frac{d\pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}(c(t))}{dt} \right|_{t=0} &= \left. \frac{d}{dt} \frac{2c(t)}{1 - K\|c(t)\|^2} \right|_{t=0} \\
 &= \frac{2v(1 - K\|x\|^2) + 4K \langle x, v \rangle x}{(1 - K\|x\|^2)^2} \\
 &= \frac{2}{(1 - K\|x\|^2)} v + \frac{4K \langle x, v \rangle}{(1 - K\|x\|^2)^2} x.
 \end{aligned} \tag{B.128}$$

Homomorphism. The homomorphism w.r.t. the scalar product can be readily obtained by the Riemannian isometry. Therefore, we first address it before proceeding to the addition. For simplicity, we denote $\phi = \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}$.

Scalar product. As shown by Ungar [200], the geodesics under the Beltrami–Klein and Poincaré ball models are

$$\begin{aligned}
 \gamma_{\phi(x) \rightarrow \phi(y)}^{\mathbb{K}}(t) &= \phi(x) \oplus_{\mathbb{E}} t \odot_{\mathbb{E}} (-\phi(x) \oplus_{\mathbb{E}} \phi(y)), \\
 \gamma_{x \rightarrow y}^{\mathbb{P}}(t) &= x \oplus_{\mathbb{M}} t \odot_{\mathbb{M}} (-x \oplus_{\mathbb{M}} y).
 \end{aligned} \tag{B.129}$$

Here, we use the fact that the gyro inverses in the Möbius and Einstein gyrovector spaces are exactly the familiar vector inverse. The above geodesics satisfy

$$\phi(x \oplus_{\mathbb{M}} t \odot_{\mathbb{M}} (-x \oplus_{\mathbb{M}} y)) = \phi(x) \oplus_{\mathbb{E}} t \odot_{\mathbb{E}} (-\phi(x) \oplus_{\mathbb{E}} \phi(y)). \tag{B.130}$$

The above comes from $\phi(\gamma_{x \rightarrow y}^{\mathbb{P}}(t)) = \gamma_{\phi(x) \rightarrow \phi(y)}^{\mathbb{K}}(t)$. In particular, the geodesic starting from the identity element yields

$$\phi(t \odot_{\mathbb{M}} x) = \phi(\gamma_{\mathbf{0} \rightarrow x}^{\mathbb{P}}(t)) = \gamma_{\phi(\mathbf{0}) \rightarrow \phi(x)}^{\mathbb{K}}(t) \stackrel{(1)}{=} t \odot_{\mathbb{E}} \phi(x), \tag{B.131}$$

where (1) comes from $\phi(\mathbf{0}) = \mathbf{0}$. The above holds for all $x \in \mathbb{P}_K^n$ and $\forall t \in \mathbb{R}$, as every hyperbolic geometry is geodesically complete [131, p. 139].

Addition. We first expand the LHS of Eq. (3.69). Inspired by Mao et al. [147, Eqs. 73–76], we express the Möbius addition as $x \oplus_M y = \frac{Bx+Cy}{A}$ by denoting

$$\begin{aligned} A &= 1 - 2K\langle x, y \rangle + K^2\|x\|^2\|y\|^2, \\ B &= 1 - 2K\langle x, y \rangle - K\|y\|^2, \\ C &= 1 + K\|x\|^2. \end{aligned} \tag{B.132}$$

Then, the Möbius addition is

$$\begin{aligned} \phi(x \oplus_M y) &= \frac{2\frac{Bx+Cy}{A}}{1 - K\left\|\frac{Bx+Cy}{A}\right\|^2} \\ &= \frac{2ABx + 2ACy}{A^2 - K\|Bx + Cy\|^2} \\ &= \frac{2ABx + 2ACy}{A^2 - B^2K\|x\|^2 - C^2K\|y\|^2 - 2BCK\langle x, y \rangle}. \end{aligned} \tag{B.133}$$

Inspired by Mao et al. [147, Eq. 77], we denote $\|x\| = a$, $\|y\| = b$, and $\langle x, y \rangle = ab\cos(\theta)$. In this way, we can resort to the symbolic computation package `SymPy` [150] for the heavy algebra computation, which brings

$$\begin{aligned} \phi(x \oplus_M y) &= \frac{2(-2Kab\cos(\theta) - Kb^2 + 1)}{K^2a^2b^2 - Ka^2 - 4Kab\cos(\theta) - Kb^2 + 1}x \\ &\quad + \frac{2Ka^2 + 2}{K^2a^2b^2 - Ka^2 - 4Kab\cos(\theta) - Kb^2 + 1}y. \end{aligned} \tag{B.134}$$

Now we turn to the RHS of Eq. (3.69). For any $u, v \in \mathbb{K}_K^n$, the Einstein addition can be rewritten as

$$\begin{aligned} u \oplus_E v &= \frac{1}{1 - K\langle u, v \rangle} \left(u + \frac{1}{\gamma_u}v - K\frac{\gamma_u}{1 + \gamma_u}\langle u, v \rangle u \right) \\ &= \frac{1 - K\frac{\gamma_u}{1 + \gamma_u}\langle u, v \rangle}{1 - K\langle u, v \rangle}u + \frac{1}{\gamma_u(1 - K\langle u, v \rangle)}v. \end{aligned} \tag{B.135}$$

The gamma factor and inner product under isometry can be rewritten as

$$\begin{aligned}
 \gamma_{\phi(x)} &= \frac{1}{\sqrt{1 + K\|\phi(x)\|^2}} \\
 &= \frac{1}{\sqrt{1 + K\left(\frac{2}{1-K\|x\|^2}\right)^2\|x\|^2}} \\
 &= \frac{1 - K\|x\|^2}{\sqrt{(1 - K\|x\|^2)^2 + 4K\|x\|^2}} \\
 &\stackrel{(1)}{=} \frac{1 - K\|x\|^2}{1 + K\|x\|^2}, \\
 \langle \phi(x), \phi(y) \rangle &= \frac{4}{(1 - K\|x\|^2)(1 - K\|y\|^2)} \langle x, y \rangle.
 \end{aligned} \tag{B.136}$$

Here, (1) comes from $\|x\|^2 < -\frac{1}{K} \Rightarrow K\|x\|^2 + 1 > 0$. Putting the above into Eq. (B.135) and following the same notation as Eq. (B.134), we can obtain the following by SymPy:

$$\begin{aligned}
 \phi(x) \oplus_E \phi(y) &= \frac{2 \cdot (2Kab \cos(\theta) + Kb^2 - 1)}{4Kab \cos(\theta) - (Ka^2 - 1)(Kb^2 - 1)} x \\
 &\quad + \frac{-2Ka^2 - 2}{4Kab \cos(\theta) - (Ka^2 - 1)(Kb^2 - 1)} y,
 \end{aligned} \tag{B.137}$$

which is clearly equal to Eq. (B.134). $\square$

B.3.18 Proof of Thm. 98

Proof. For simplicity, we denote $\phi = \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}$. The homomorphism and bijection of ϕ imply the homomorphism of its inverse ϕ^{-1} . Also note that $\phi(\mathbf{0}) = \phi^{-1}(\mathbf{0}) = \mathbf{0}$. This yields

$$\begin{aligned}
 x \oplus_E y &= \phi(\phi^{-1}(x) \oplus_M \phi^{-1}(y)) \\
 &\stackrel{(1)}{=} \phi\left(\text{Exp}_{\phi^{-1}(x)}^{\mathbb{P}}\left(\text{PT}_{\mathbf{0} \rightarrow \phi^{-1}(x)}^{\mathbb{P}} \text{Log}_{\mathbf{0}}^{\mathbb{P}}(\phi^{-1}(y))\right)\right) \\
 &\stackrel{(2)}{=} \text{Exp}_x^{\mathbb{K}}\left(\text{PT}_{\mathbf{0} \rightarrow x}^{\mathbb{K}} \text{Log}_{\mathbf{0}}^{\mathbb{K}}(y)\right).
 \end{aligned} \tag{B.138}$$

The above comes from the following.

(1) For any Poincaré vectors $u, w \in \mathbb{P}_K^n$, the following holds:

$$\text{PT}_{\mathbf{0} \rightarrow u}^{\mathbb{P}}(v) = \text{Log}_u^{\mathbb{P}}(u \oplus_M \text{Exp}_{\mathbf{0}}^{\mathbb{P}}(v)) \Rightarrow u \oplus_M w = \text{Exp}_u^{\mathbb{P}}(\text{PT}_{\mathbf{0} \rightarrow u}^{\mathbb{P}}(\text{Log}_{\mathbf{0}}^{\mathbb{P}}(w))), \tag{B.139}$$

where $v = \text{Log}_{\mathbf{0}}^{\mathbb{P}}(w)$ and the LHS comes from Ganea et al. [76, Thm. 4];

(2) The second identity follows from the isometry:

$$\begin{aligned}\mathrm{Exp}_x^{\mathbb{P}}(v) &= \phi^{-1} \left(\mathrm{Exp}_{\phi(x)}^{\mathbb{K}}(\phi_{*,x}(v)) \right), \\ \mathrm{Log}_x^{\mathbb{P}}(y) &= \phi_{*,x}^{-1} \left(\mathrm{Log}_{\phi(x)}^{\mathbb{K}}(\phi(y)) \right), \\ \mathrm{PT}_{x \rightarrow y}^{\mathbb{P}}(v) &= \phi_{*,y}^{-1} \left(\mathrm{PT}_{\phi(x) \rightarrow \phi(y)}^{\mathbb{K}}(\phi_{*,x}(v)) \right).\end{aligned}\tag{B.140}$$

Similar to the gyro addition, the isometry of ϕ implies the following with respect to the gyro scalar product:

$$\begin{aligned}t \odot_{\mathrm{E}} x &= \phi \left(t \odot_{\mathrm{M}} \phi^{-1}(x) \right) \\ &\stackrel{(1)}{=} \phi \left(\mathrm{Exp}_{\mathbf{0}}^{\mathbb{P}} \left(t \mathrm{Log}_{\mathbf{0}}^{\mathbb{P}}(\phi^{-1}(x)) \right) \right) \\ &\stackrel{(2)}{=} \mathrm{Exp}_{\mathbf{0}}^{\mathbb{K}} \left(t \mathrm{Log}_{\mathbf{0}}^{\mathbb{K}}(x) \right).\end{aligned}\tag{B.141}$$

The above comes from the following.

(1) Ganea et al. [76, Lem. 3];

(2) The isometry of ϕ .

□

B.3.19 Proof of Thm. 99

Proof. Following Thm. 98, we denote $\phi = \pi_{\mathbb{P}_K^n \rightarrow \mathbb{K}_K^n}$ and $\psi = \pi_{\mathbb{K}_K^n \rightarrow \mathbb{P}_K^n}$. The results can be obtained by the properties of isometry and isomorphism of ϕ .

Riemannian exponential and logarithmic maps at the zero vector. First, we recall that the expressions of $\mathrm{Exp}_{\mathbf{0}}^{\mathbb{P}}(v)$ and $\mathrm{Log}_{\mathbf{0}}^{\mathbb{P}}(v)$ under the Poincaré ball model are exactly those in Eqs. (3.72) and (3.73) [76, Eq. 13].

By Riemannian isometry, we have the following:

$$\begin{aligned}
 \text{Exp}_{\mathbf{0}}^{\mathbb{K}}(v) &= \phi \left(\text{Exp}_{\mathbf{0}}^{\mathbb{P}}(\psi_{*,\mathbf{0}}(v)) \right) \\
 &= \phi \left(\tanh \left(\frac{\sqrt{-K}}{2} \|v\| \right) \frac{v}{\sqrt{-K} \|v\|} \right) \\
 &= \frac{2}{1 - K \|v\|^2 \left(\frac{\tanh \left(\frac{\sqrt{-K}}{2} \|v\| \right)}{\sqrt{-K} \|v\|} \right)^2} \frac{\tanh \left(\frac{\sqrt{-K}}{2} \|v\| \right)}{\sqrt{-K} \|v\|} v \\
 &= \frac{2 \tanh \left(\frac{\sqrt{-K}}{2} \|v\| \right)}{1 + \tanh \left(\frac{\sqrt{-K}}{2} \|v\| \right)^2} \frac{v}{\sqrt{-K} \|v\|} \\
 &\stackrel{(1)}{=} \tanh(\sqrt{-K} \|v\|) \frac{v}{\sqrt{-K} \|v\|},
 \end{aligned} \tag{B.142}$$

where (1) comes from $\tanh(2x) = \frac{2 \tanh x}{1 + \tanh^2 x}$.

As the inverse of $\text{Exp}_{\mathbf{0}}^{\mathbb{K}}(v)$, $\text{Log}_{\mathbf{0}}^{\mathbb{K}}(x)$, therefore, shares the expression with its counterpart under the Poincaré ball model. Here, we use the properties of isometry to further validate this result:

$$\begin{aligned}
 \text{Log}_{\mathbf{0}}^{\mathbb{K}}(x) &= \phi_{*,0} \left(\text{Log}_{\mathbf{0}}^{\mathbb{P}}(\psi(x)) \right) \\
 &= 2 \tanh^{-1} \left(\frac{\sqrt{-K} \|x\|}{1 + \sqrt{1 + K \|x\|^2}} \right) \frac{x}{\sqrt{-K} \|x\|}.
 \end{aligned} \tag{B.143}$$

We only need to show

$$2 \tanh^{-1} \left(\frac{\sqrt{-K} \|x\|}{1 + \sqrt{1 + K \|x\|^2}} \right) = \tanh^{-1}(\sqrt{-K} \|x\|). \tag{B.144}$$

We denote $a = \sqrt{-K} \|x\| < 1$. Both sides are

$$\begin{aligned}
 \text{LHS: } 2 \tanh^{-1} \left(\frac{a}{1 + \sqrt{1 - a^2}} \right) &= \ln \left(\frac{1 + \frac{a}{1 + \sqrt{1 - a^2}}}{1 - \frac{a}{1 + \sqrt{1 - a^2}}} \right) = \ln \left(\frac{1 + \sqrt{1 - a^2} + a}{1 + \sqrt{1 - a^2} - a} \right), \\
 \text{RHS: } \tanh^{-1}(a) &= \ln \left(\frac{\sqrt{1 + a}}{\sqrt{1 - a}} \right) = \ln \left(\frac{\sqrt{1 - a^2}}{1 - a} \right).
 \end{aligned} \tag{B.145}$$

Note that the below equation holds:

$$\frac{1 + \sqrt{1 - a^2} + a}{1 + \sqrt{1 - a^2} - a} = \frac{\sqrt{1 - a^2}}{1 - a}. \quad (\text{B.146})$$

Geodesic distances.

$$\begin{aligned} d^{\mathbb{K}}(x, y) &\stackrel{(1)}{=} d^{\mathbb{P}}(\psi(x), \psi(y)) \\ &= \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \|\psi(x) \oplus_{\mathbb{M}} \psi(y)\| \right) \\ &\stackrel{(2)}{=} \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \|\psi(-x \oplus_{\mathbb{E}} y)\| \right) \\ &= \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \frac{\|-x \oplus_{\mathbb{E}} y\|}{1 + \sqrt{1 + K \|-x \oplus_{\mathbb{E}} y\|^2}} \right), \end{aligned} \quad (\text{B.147})$$

where (1) comes from the isometry, while (2) comes from the isomorphism.

Exponential maps.

$$\begin{aligned} \text{Exp}_x^{\mathbb{K}}(v) &\stackrel{(1)}{=} \psi^{-1} \left(\text{Exp}_{\psi(x)}^{\mathbb{P}}(\psi_{*,x}(v)) \right) \\ &\stackrel{(2)}{=} \psi^{-1} \left(\psi(x) \oplus_{\mathbb{M}} \text{Exp}_{\mathbf{0}} \left(\frac{\lambda_{\psi(x)}^K}{2} \psi_{*,x}(v) \right) \right) \\ &\stackrel{(3)}{=} x \oplus_{\mathbb{E}} \psi^{-1} \left(\text{Exp}_{\mathbf{0}} \left(\frac{\lambda_{\psi(x)}^K}{2} \psi_{*,x}(v) \right) \right) \\ &\stackrel{(4)}{=} x \oplus_{\mathbb{E}} \text{Exp}_{\mathbf{0}} \left(\lambda_{\psi(x)}^K \psi_{*,x}(v) \right). \end{aligned} \quad (\text{B.148})$$

The above comes from the following.

(1) The isometry of ψ ;

(2) $\text{Exp}_x^{\mathbb{P}}(v) = x \oplus_{\mathbb{M}} \text{Exp}_{\mathbf{0}} \left(\frac{\lambda_x^K}{2} v \right)$;

(3) The isomorphism of ψ ;

(4)

$$\begin{aligned} \psi^{-1} \circ \text{Exp}_{\mathbf{0}}(v) &= \psi^{-1} \circ \text{Exp}_{\mathbf{0}}^{\mathbb{P}}(v) \\ &= \psi^{-1} \circ \psi \circ \text{Exp}_{\mathbf{0}}^{\mathbb{K}}(2v) \\ &= \text{Exp}_{\mathbf{0}}(2v). \end{aligned} \quad (\text{B.149})$$

It remains to calculate $\lambda_{\psi(x)}^K \psi_{*,x}(v)$:

$$\begin{aligned}
 \lambda_{\psi(x)}^K &= \frac{2}{1 + K \frac{\|x\|^2}{(1 + \sqrt{1 + K\|x\|^2})^2}} \\
 &= \frac{2 \left(1 + \sqrt{1 + K\|x\|^2}\right)^2}{\left(1 + \sqrt{1 + K\|x\|^2}\right)^2 + K\|x\|^2} \\
 &= \frac{2 \left(1 + \sqrt{1 + K\|x\|^2}\right)^2}{2 + 2\sqrt{1 + K\|x\|^2} + 2K\|x\|^2} \\
 &= \frac{\left(1 + \sqrt{1 + K\|x\|^2}\right)^2}{1 + \sqrt{1 + K\|x\|^2} + K\|x\|^2} \\
 &= \frac{\left(1 + \sqrt{1 + K\|x\|^2}\right)}{\sqrt{1 + K\|x\|^2}},
 \end{aligned} \tag{B.150}$$

$$\begin{aligned}
 \lambda_{\psi(x)}^K \psi_{*,x}(v) &= \lambda_{\psi(x)}^K \left(\frac{1}{1 + \sqrt{1 + K\|x\|^2}} v - \frac{K \langle x, v \rangle}{\left(1 + \sqrt{1 + K\|x\|^2}\right)^2 \sqrt{1 + K\|x\|^2}} x \right) \\
 &= \frac{1}{\sqrt{1 + K\|x\|^2}} v - \frac{K \langle x, v \rangle}{\left(1 + \sqrt{1 + K\|x\|^2}\right) (1 + K\|x\|^2)} x.
 \end{aligned} \tag{B.151}$$

Logarithmic maps.

$$\begin{aligned}
 \text{Log}_x^{\mathbb{K}}(y) &\stackrel{(1)}{=} \phi_{*,\psi(x)} (\text{Log}_{\psi(x)}^{\mathbb{P}} \psi(y)) \\
 &\stackrel{(2)}{=} \phi_{*,\psi(x)} \left(\frac{2}{\lambda_{\psi(x)}^K} \text{Log}_{\mathbf{0}} (-\psi(x) \oplus_{\mathbf{M}} \psi(y)) \right) \\
 &\stackrel{(3)}{=} \frac{2}{\lambda_{\psi(x)}^K} \phi_{*,\psi(x)} (\text{Log}_{\mathbf{0}} (-\psi(x) \oplus_{\mathbf{M}} \psi(y))) \\
 &\stackrel{(4)}{=} \frac{1}{\lambda_{\psi(x)}^K} \phi_{*,\psi(x)} (\text{Log}_{\mathbf{0}} (-x \oplus_{\mathbf{E}} y)).
 \end{aligned} \tag{B.152}$$

The above comes from the following.

- (1) The isometry of ψ ;
- (2) $\text{Log}_x^{\mathbb{P}}(y) = \frac{2}{\lambda_x^K} \text{Log}_{\mathbf{0}} (-x \oplus_{\mathbf{M}} y)$;
- (3) The linearity of differential maps;

(4)

$$\begin{aligned}
 \text{Log}_0(-\psi(x) \oplus_M \psi(y)) &= \text{Log}_0^{\mathbb{P}}(-\psi(x) \oplus_M \psi(y)) \\
 &= \frac{1}{2} \text{Log}_0^{\mathbb{K}}(\psi^{-1}(-\psi(x) \oplus_M \psi(y))) \\
 &= \frac{1}{2} \text{Log}_0^{\mathbb{K}}(-x \oplus_E y).
 \end{aligned} \tag{B.153}$$

□

Remark 192. Mao et al. [147, Thm. 9] extended the Möbius matrix-vector multiplication [76, Lem. 6] to the Einstein space of the Beltrami–Klein model under $K = -1$, namely $\text{Exp}_0(M \text{Log}_0(x))$. Although their presented formulations are different from the Möbius one, this theorem indicates that matrix-vector multiplications over these two spaces are identical under any negative curvature. The equality can also be readily observed from Eqs. (B.145) and (B.146).

B.4 SPD Multinomial Logistic Regression

B.4.1 Proof of Thm. 104

This claim can be proven either by definition [197, Def. 9.1] or by the constant rank level set theorem [197, Thm. 11.2]. We focus on the latter.

Proof. Consider any $P \in \mathcal{S}_{++}^n$ and $A \in T_P \mathcal{S}_{++}^n \setminus \{0\}$. Define the function

$$f : \mathcal{S}_{++}^n \rightarrow \mathbb{R}, \quad S \mapsto \langle \text{Log}_P S, A \rangle_P. \tag{B.154}$$

For the SPD hyperplane $\tilde{H}_{A,P}$, we have $\tilde{H}_{A,P} = f^{-1}(0)$. By assumption, $\text{Log}_P : \mathcal{S}_{++}^n \rightarrow T_P \mathcal{S}_{++}^n$ is a global diffeomorphism, and f is therefore well-defined. We can rewrite f as a composition, *i.e.*, $f = h \circ \text{Log}_P$, where $h(\cdot) = \langle \cdot, A \rangle_P$ is a linear map.

Since Log_P is a diffeomorphism and $h(\cdot)$ is a nonzero linear map, the rank of f is globally constant. Therefore, there exists a neighborhood, *e.g.*, the whole SPD manifold, of $f^{-1}(0)$ where the rank of f is constant. According to the constant rank level set theorem [197, Thm. 11.2], we can obtain the claim. □

B.4.2 Proof of Thm. 105

Proof. By Thm. 149, we have

$$\langle \text{Log}_P Q, A \rangle_P = \left\langle \phi_{*,P} \phi_{*,\phi(P)}^{-1} (\phi(Q) - \phi(P)), \phi_{*,P} A \right\rangle \quad (\text{B.155})$$

$$= \langle \phi(Q) - \phi(P), \phi_{*,P} A \rangle. \quad (\text{B.156})$$

Therefore, due to the isometry of ϕ , the SPD hyperplane $\tilde{H}_{A_k, P_k}$ corresponds to the Euclidean hyperplane

$$H_{\phi_{*,P_k}(A_k), \phi(P_k)}. \quad (\text{B.157})$$

Furthermore, the distance to the margin hyperplane is equivalent to

$$\inf_{\phi(Q)} \|\phi(S) - \phi(Q)\|_F \quad (\text{B.158})$$

$$\text{s.t. } \langle \phi(Q) - \phi(P_k), \phi_{*,P_k} A_k \rangle = 0. \quad (\text{B.159})$$

The problem above is the familiar Euclidean distance from a point to a hyperplane. By simple computation, one can obtain the result. $\square$

B.4.3 Proof of Thm. 106

Proof. For simplicity, we abbreviate $\odot_\phi$ and g^ϕ as $\odot$ and g . By abuse of notation, we further denote $Q \odot P_\odot^{-1}$ as QP^{-1} , where $P_\odot^{-1}$ is the inverse of P under $\odot$. According to Thm. 149, $(\mathcal{S}_{++}^n, \odot)$ is an abelian group and g is a bi-invariant Riemannian metric. By Lin [137, Lem. 6], any parallel transport can be expressed as a differential of left translation,

$$\text{PT}_{P \rightarrow Q} = (L_{QP^{-1}})_{*,P}, \quad \forall P, Q \in \mathcal{S}_{++}^n. \quad (\text{B.160})$$

$\square$

B.4.4 Proof of Thm. 107

Proof. By Eq. (6.9), parallel transport under a pullback Euclidean metric is path-independent and satisfies

$$\text{PT}_{Q \rightarrow P}(V) = \phi_{*,\phi(P)}^{-1} (\phi_{*,Q}(V)). \quad (\text{B.161})$$

For any $\tilde{A}_{1,k} \in T_{Q_1} \mathcal{S}_{++}^n$, define

$$\tilde{A}_{2,k} = \phi_{*,\phi(Q_2)}^{-1} \left(\phi_{*,Q_1} \left(\tilde{A}_{1,k} \right) \right) \in T_{Q_2} \mathcal{S}_{++}^n. \quad (\text{B.162})$$

Since ϕ is a diffeomorphism, its differential at every point is a linear isomorphism. Hence,

$$\begin{aligned} \text{PT}_{Q_2 \rightarrow P_k} \left(\tilde{A}_{2,k} \right) &= \phi_{*,\phi(P_k)}^{-1} \left(\phi_{*,Q_2} \left(\tilde{A}_{2,k} \right) \right) \\ &= \phi_{*,\phi(P_k)}^{-1} \left(\phi_{*,Q_1} \left(\tilde{A}_{1,k} \right) \right) \\ &= \text{PT}_{Q_1 \rightarrow P_k} \left(\tilde{A}_{1,k} \right). \end{aligned} \quad (\text{B.163})$$

Moreover, if $\tilde{B}_{2,k} \in T_{Q_2} \mathcal{S}_{++}^n$ satisfies the same equality, applying ϕ_{*,P_k} to both sides gives

$$\phi_{*,Q_2} \left(\tilde{B}_{2,k} \right) = \phi_{*,Q_1} \left(\tilde{A}_{1,k} \right). \quad (\text{B.164})$$

The invertibility of ϕ_{*,Q_2} then yields $\tilde{B}_{2,k} = \tilde{A}_{2,k}$, proving uniqueness. $\square$

B.4.5 Proof of Thm. 108

Proof.

$$A_k = \text{PT}_{I \rightarrow P_k}(\tilde{A}_k) \quad (\text{B.165})$$

$$= \phi_{*,\phi(P_k)}^{-1} \left(\phi_{*,I}(\tilde{A}_k) \right). \quad (\text{B.166})$$

One can obtain the result by putting Eq. (B.166) into Eq. (4.10). $\square$

B.4.6 Proof of Thm. 109

Proof. Denoting the matrix power as $P_\theta : \mathcal{S}_{++}^n \rightarrow \mathcal{S}_{++}^n$, we have

$$P_\theta(I) = I, \quad (\text{B.167})$$

$$(P_\theta)_{*,I}(A) = \theta A, \quad \forall A \in T_I \mathcal{S}_{++}^n. \quad (\text{B.168})$$

Next, we prove the two cases separately.

(α, β) -LEM. We define the following map:

$$\psi^{\text{LEM}} = f \circ \log, \quad (\text{B.169})$$

where $f : \mathcal{S}^n \rightarrow \mathcal{S}^n$ is the linear isometry between the standard Frobenius inner product

and the $O(n)$ -invariant inner product $\langle \cdot, \cdot \rangle^{(\alpha, \beta)}$. Then ψ^{LEM} pulls back the standard Euclidean metric on $\mathcal{S}^n$ to (α, β) -LEM on $\mathcal{S}_{++}^n$. Putting Eqs. (B.168) and (B.169) into Eq. (4.12), we have

$$\begin{aligned} & \exp \left(\left\langle \psi^{\text{LEM}}(S) - \psi^{\text{LEM}}(P_k), \psi_{*,I}^{\text{LEM}}(\tilde{A}_k) \right\rangle \right) \\ &= \exp \left[\left\langle f(\log(S) - \log(P_k)), f(\tilde{A}_k) \right\rangle \right] \\ &= \exp \left[\left\langle \log(S) - \log(P_k), \tilde{A}_k \right\rangle^{(\alpha, \beta)} \right], \end{aligned} \quad (\text{B.170})$$

where the last equality follows from the linearity and isometry property of f , with $f = f_*$.

θ -LCM. We denote

$$\psi^{\text{LCM}} = \text{Dlog} \circ \text{Chol} \circ \text{P}_\theta. \quad (\text{B.171})$$

Then ψ^{LCM} pulls back the Euclidean metric $\frac{1}{\theta^2} g^{\text{E}}$ on the Euclidean space LT^n of lower triangular matrices to θ -LCM on $\mathcal{S}_{++}^n$. The differential of Cholesky decomposition is presented by Lin [137, Prop. 4], while the differential of Dlog is obtained as the natural-logarithm specialization of Thm. 154. Then, simple computations show that

$$\psi_{*,I}^{\text{LCM}}(A) = \theta \left(\lfloor A \rfloor + \frac{1}{2} \mathbb{D}(A) \right), \quad \forall A \in T_I \mathcal{S}_{++}^n. \quad (\text{B.172})$$

Putting Eqs. (B.171) and (B.172) into Eq. (4.12), we can obtain the result. $\square$

B.4.7 Proof of Thm. 111

To prove Thm. 111, we first present two lemmas about the general cases under pullback Euclidean metrics.

One can observe that Eqs. (4.12) and (4.13) are very similar to those of a Euclidean MLR. However, since ϕ is normally nonlinear and P_k is an SPD parameter, Eq. (4.12) cannot hastily be identified with a Euclidean MLR. Under some special circumstances, SPD MLR can be reduced to the familiar Euclidean MLR. To show this result, we first specialize the RSGD strategy reviewed in Sec. 2.7 to pullback Euclidean metrics. In ambient-gradient notation, the required update is

$$W_{t+1} = \text{Exp}_{W_t}(-\gamma_t \Pi_{W_t}(\nabla_{W_t} f)), \quad (\text{B.173})$$

where Π_{W_t} denotes the projection mapping the Euclidean gradient $\nabla_{W_t} f$ to the Rie-

mannian gradient, and γ_t denotes the learning rate. We have already obtained the formula for the Riemannian exponential map in Eq. (6.7). We proceed to formulate Π .

Lemma 193. *For a smooth function $f : \mathcal{S}_{++}^n \rightarrow \mathbb{R}$ on $\mathcal{S}_{++}^n$ endowed with any kind of pullback Euclidean metric, the projection map $\Pi_P : \mathcal{S}^n \rightarrow T_P \mathcal{S}_{++}^n$ at $P \in \mathcal{S}_{++}^n$ is*

$$\Pi_P(\nabla_P f) = \phi_{*,\phi(P)}^{-1} \left(\phi_{*,\phi(P)}^{-*}(\nabla_P f) \right), \quad (\text{B.174})$$

where $\phi_{*,\phi(P)}^{-*} := \left(\phi_{*,\phi(P)}^{-1} \right)^* : T_P \mathcal{S}_{++}^n \rightarrow T_{\phi(P)} \mathcal{S}^n$ is the Frobenius adjoint of $\phi_{*,\phi(P)}^{-1} : T_{\phi(P)} \mathcal{S}^n \rightarrow T_P \mathcal{S}_{++}^n$. Specifically, for all $U \in T_P \mathcal{S}_{++}^n$ and $Z \in T_{\phi(P)} \mathcal{S}^n$, it satisfies

$$\langle U, \phi_{*,\phi(P)}^{-1}(Z) \rangle = \langle \phi_{*,\phi(P)}^{-*}(U), Z \rangle. \quad (\text{B.175})$$

Proof. Given any smooth function $f : \mathcal{S}_{++}^n \rightarrow \mathbb{R}$, denote its Riemannian gradient at P as $\text{grad}_P f \in T_P \mathcal{S}_{++}^n$.

$$\langle \text{grad}_P f, V \rangle_P = f_{*,P}(V), \quad \forall V \in T_P \mathcal{S}_{++}^n. \quad (\text{B.176})$$

Let $\nabla_P f$ denote the Euclidean gradient in the canonical chart. For any $Z \in T_{\phi(P)} \mathcal{S}^n$, set $V = \phi_{*,\phi(P)}^{-1}(Z)$. By Eq. (6.5) and the definition of the Frobenius adjoint, we have

$$\begin{aligned} \langle \phi_{*,P}(\text{grad}_P f), Z \rangle &= \left\langle \text{grad}_P f, \phi_{*,\phi(P)}^{-1}(Z) \right\rangle_P \\ &= f_{*,P} \left(\phi_{*,\phi(P)}^{-1}(Z) \right) \\ &= \left\langle \nabla_P f, \phi_{*,\phi(P)}^{-1}(Z) \right\rangle \\ &= \left\langle \phi_{*,\phi(P)}^{-*}(\nabla_P f), Z \right\rangle. \end{aligned} \quad (\text{B.177})$$

Since Z is arbitrary, the nondegeneracy of the Frobenius inner product gives

$$\phi_{*,P}(\text{grad}_P f) = \phi_{*,\phi(P)}^{-*}(\nabla_P f). \quad (\text{B.178})$$

Applying $\phi_{*,\phi(P)}^{-1}$ to both sides yields

$$\text{grad}_P f = \phi_{*,\phi(P)}^{-1} \left(\phi_{*,\phi(P)}^{-*}(\nabla_P f) \right). \quad (\text{B.179})$$

By definition, $\Pi_P(\nabla_P f) = \text{grad}_P f$, which yields Eq. (B.174). $\square$

We can describe the special case mentioned above with this lemma.

Lemma 194. *Suppose the differential map $\phi_{*,I}$ is the identity map, and P_k in Eq. (4.12) is optimized by pullback-Euclidean-metric-based RSGD. Then Eq. (4.12) can be reduced to a Euclidean MLR in the codomain of ϕ updated by Euclidean SGD.*

Proof. Define a Euclidean MLR in the codomain of ϕ as

$$p(y = k \mid S) \propto \exp \left(\langle \phi(S) - \bar{P}_k, \bar{A}_k \rangle \right), \quad (\text{B.180})$$

where $\bar{P}_k, \bar{A}_k \in \mathcal{S}^n$. We call this classifier ϕ -EMLR.

Define the SPD MLR under the pullback Euclidean metric induced by ϕ as

$$p(y = k \mid S) \propto \exp \left(\left\langle \phi(S) - \phi(P_k), \tilde{A}_k \right\rangle \right), \quad (\text{B.181})$$

where $P_k \in \mathcal{S}_{++}^n$ and $\tilde{A}_k \in \mathcal{S}^n$.

Suppose the SPD MLR and ϕ -EMLR satisfy $\bar{P}_k = \phi(P_k)$. Other settings of the network are all the same, indicating that the Euclidean gradients satisfy

$$\frac{\partial L}{\partial \bar{P}_k} = \frac{\partial L}{\partial \phi(P_k)}. \quad (\text{B.182})$$

The update of $\bar{P}_k$ in the ϕ -EMLR is

$$\bar{P}'_k = \bar{P}_k - \gamma \frac{\partial L}{\partial \bar{P}_k}. \quad (\text{B.183})$$

The update of P_k in the SPD MLR is

$$P'_k = \text{Exp}_{P_k} \left(-\gamma \Pi_{P_k} \left(\frac{\partial L}{\partial P_k} \right) \right) \quad (\text{B.184})$$

$$= \phi^{-1} \left(\phi(P_k) - \gamma \phi_{*,P_k}^* \frac{\partial L}{\partial P_k} \right). \quad (\text{B.185})$$

Therefore, $\phi(P'_k)$ satisfies

$$\phi(P'_k) = \phi(P_k) - \gamma \phi_{*,P_k}^{-*} \frac{\partial L}{\partial P_k} \quad (\text{B.186})$$

$$= \phi(P_k) - \gamma \phi_{*,P_k}^{-*} \phi_{*,P_k}^* \frac{\partial L}{\partial \phi(P_k)} \quad (\text{B.187})$$

$$= \phi(P_k) - \gamma \frac{\partial L}{\partial \phi(P_k)} \quad (\text{B.188})$$

$$= \bar{P}'_k. \quad (\text{B.189})$$

Eq. (B.187) comes from the Euclidean chain rule of the differential. Let $Y = \phi(X)$. Then we have

$$\frac{\partial L}{\partial Y} : dY = \frac{\partial L}{\partial Y} : \phi_{*,X} dX = \phi_{*,X}^* \frac{\partial L}{\partial Y} : dX, \quad (\text{B.190})$$

where $:$ denotes the Frobenius inner product.

The equivalence of $\bar{A}_k$ and $\tilde{A}_k$ is obvious. By mathematical induction, the claim can be proven. $\square$

Proof. The claim follows directly from Thm. 194. $\square$

B.5 Riemannian Multinomial Logistic Regression

B.5.1 Proof of Thm. 113

Proof. Let us first solve Y^* in Eq. (4.22), which is the solution to the following constrained optimization problem:

$$\max_Y \left(\frac{\langle \text{Log}_P Y, \text{Log}_P S \rangle_P}{\|\text{Log}_P Y\|_P \|\text{Log}_P S\|_P} \right) \quad \text{s.t.} \langle \text{Log}_P Y, \tilde{A} \rangle_P = 0. \quad (\text{B.191})$$

Note that Eq. (B.191) is well-defined due to the existence of the Riemannian logarithm. Although Eq. (B.191) is normally non-convex, Eq. (B.191) and Eq. (4.22) can be reduced to a Euclidean problem:

$$\max_{\tilde{Y}} \frac{\langle \tilde{Y}, \tilde{S} \rangle_P}{\|\tilde{Y}\|_P \|\tilde{S}\|_P} \quad \text{s.t.} \langle \tilde{Y}, \tilde{A} \rangle_P = 0, \quad (\text{B.192})$$

$$d(S, \tilde{H}_{\tilde{A},P}) = \sin(\angle SPY^*) \|\tilde{S}\|_P, \quad (\text{B.193})$$

where $\tilde{Y} = \text{Log}_P Y$ and $\tilde{S} = \text{Log}_P S$.

Let us first discuss Eq. (B.192). Denote the solution of Eq. (B.192) as $\tilde{Y}^*$. Note that $\tilde{Y}^*$ is not necessarily unique. Note that Exp_P is only well-defined locally. More precisely, Exp_P is well-defined in an open ball $B_\epsilon(0)$ centered at $0 \in T_P \mathcal{M}$. Therefore, $\tilde{Y}^*$ might not be in $B_\epsilon(0)$. In this case, we can scale $\tilde{Y}^*$ into $B_\epsilon(0)$, and the scaled $\tilde{Y}^*$ is still the maximizer of Eq. (B.192). Therefore, without loss of generality, we assume $\tilde{Y}^* \in B_\epsilon(0)$.

Putting $\tilde{Y}^*$ into Eq. (B.193), Eq. (B.193) is reduced to the distance to the hyperplane

$\langle \tilde{Y}, \tilde{A} \rangle_P = 0$ in the Euclidean space $(T_P \mathcal{M}, \langle \cdot, \cdot \rangle_P)$, which has a closed-form solution:

$$d(S, \tilde{H}_{\tilde{A}, P}) = \frac{|\langle \tilde{S}, \tilde{A} \rangle_P|}{\|\tilde{A}\|_P} \quad (\text{B.194})$$

$$= \frac{|\langle \text{Log}_P S, \tilde{A} \rangle_P|}{\|\tilde{A}\|_P}. \quad (\text{B.195})$$

□

B.5.2 Proof of Thm. 114

Proof. Putting the margin distance (Eq. (4.25)) into Eq. (4.18), we have the following:

$$\begin{aligned} p(y = k \mid S) &\propto \exp \left(\text{sign}(\langle \tilde{A}_k, \text{Log}_{P_k}(S) \rangle_{P_k}) \|\tilde{A}_k\|_{P_k} d(S, \tilde{H}_{\tilde{A}_k, P_k}) \right) \\ &= \exp \left(\text{sign}(\langle \tilde{A}_k, \text{Log}_{P_k}(S) \rangle_{P_k}) \|\tilde{A}_k\|_{P_k} \frac{|\langle \text{Log}_{P_k}(S), \tilde{A}_k \rangle_{P_k}|}{\|\tilde{A}_k\|_{P_k}} \right) \\ &= \exp \left(\langle \text{Log}_{P_k} S, \tilde{A}_k \rangle_{P_k} \right). \end{aligned} \quad (\text{B.196})$$

□

B.5.3 Proof of Thm. 116

Proof. The Riemannian metric (α, β) -EM at I is

$$g_I^{(\alpha, \beta)\text{-EM}}(V, V) = \langle V, V \rangle^{(\alpha, \beta)}. \quad (\text{B.197})$$

By Thm. 186, we have the following:

$$\begin{aligned} g_P^{(\theta, \alpha, \beta)\text{-EM}}(V, V) &\xrightarrow{\theta \rightarrow 0} g_I^{(\alpha, \beta)\text{-EM}}(\log_{*, P}(V), \log_{*, P}(V)) \\ &= \langle \log_{*, P}(V), \log_{*, P}(V) \rangle^{(\alpha, \beta)} \\ &= g_P^{(\alpha, \beta)\text{-LEM}}(V, V). \end{aligned} \quad (\text{B.198})$$

□

B.5.4 Proof of Thm. 117

As the five families of metrics presented in Thm. 117 are pullback metrics, we first present a general result regarding Riemannian MLRs under pullback metrics.

Lemma 195 (Riemannian MLRs under pullback metrics). *Suppose $(\mathcal{N}, g)$ is a Riemannian manifold and $\phi : \mathcal{M} \rightarrow \mathcal{N}$ is a diffeomorphism between manifolds. The Riemannian MLR by parallel transport, obtained by combining Eqs. (4.26) and (4.27), on $\mathcal{M}$ under $\tilde{g} = \phi^*g$ can be obtained using g :*

$$p(y = k \mid S \in \mathcal{M}) \propto \exp \left[\langle \tilde{\text{Log}}_{P_k} S, \tilde{\Gamma}_{Q \rightarrow P_k} A_k \rangle_{P_k} \right] \quad (\text{B.199})$$

$$= \exp \left[\langle \text{Log}_{\phi(P_k)} \phi(S), \tilde{A}_k \rangle_{\phi(P_k)} \right], \quad (\text{B.200})$$

where $\tilde{A}_k = \Gamma_{\phi(Q) \rightarrow \phi(P_k)} \phi_{*,Q}(A_k)$ with $A_k \in T_Q \mathcal{M}$; $\tilde{\text{Log}}$ and $\tilde{\Gamma}$ are the Riemannian logarithm and parallel transport under $\tilde{g}$, respectively; and Log and Γ are their counterparts under g .

Furthermore, if $\mathcal{N}$ has a Lie group operation $\odot$, $\mathcal{M}$ could be endowed with a Lie group structure $\tilde{\odot}$ by ϕ . The Riemannian MLR by left translation, obtained by combining Eqs. (4.26) and (4.28), on $\mathcal{M}$ under $\tilde{g}$ and $\tilde{\odot}$ can be calculated using g and $\odot$:

$$p(y = k \mid S \in \mathcal{M}) \propto \exp \left[\left\langle \tilde{\text{Log}}_{P_k} S, \left(\tilde{L}_{\tilde{R}_k} \right)_{*,Q} A_k \right\rangle_{P_k} \right] \quad (\text{B.201})$$

$$= \exp \left[\langle \text{Log}_{\phi(P_k)} \phi(S), \tilde{A}_k \rangle_{\phi(P_k)} \right], \quad (\text{B.202})$$

where $\tilde{A}_k = (L_{R_k})_{*,\phi(Q)}(\phi_{*,Q}(A_k))$, $\tilde{R}_k = P_k \tilde{\odot} Q \tilde{\odot}^{-1}$, $R_k = \phi(P_k) \odot \phi(Q)^{-1}$, and $\tilde{L}_{P_k \tilde{\odot} Q \tilde{\odot}^{-1}}$ is the left translation under $\tilde{\odot}$.

Proof. Before starting, we should point out that since ϕ is a diffeomorphism, $\tilde{\odot}$ and $\tilde{g}$ are indeed well defined, and $(\mathcal{M}, \tilde{g})$ forms a Riemannian manifold and $(\mathcal{M}, \tilde{\odot})$ forms a Lie group. We first focus on the Riemannian MLR by parallel transport:

$$\begin{aligned} p(y = k \mid S \in \mathcal{M}) &\propto \exp \left(\tilde{g}_{P_k}(\tilde{\text{Log}}_{P_k} S, \tilde{\Gamma}_{Q \rightarrow P_k} A_k) \right) \\ &= \exp \left[g_{\phi(P_k)} \left(\phi_{*,P_k} \circ \phi_{*,\phi(P_k)}^{-1} \text{Log}_{\phi(P_k)} \phi(S), \phi_{*,P_k} \circ \phi_{*,\phi(P_k)}^{-1} \Gamma_{\phi(Q) \rightarrow \phi(P_k)} \phi_{*,Q}(A_k) \right) \right] \\ &= \exp \left[g_{\phi(P_k)}(\text{Log}_{\phi(P_k)} \phi(S), \Gamma_{\phi(Q) \rightarrow \phi(P_k)} \phi_{*,Q}(A_k)) \right]. \end{aligned} \quad (\text{B.203})$$

In the case of the Riemannian MLR by left translation, we first note that

$$\tilde{L}_{\tilde{R}_k} = \phi^{-1} \circ L_{\phi(P_k) \odot \phi(Q)^{-1}} \circ \phi. \quad (\text{B.204})$$

Therefore, the associated differential is

$$\left(\tilde{L}_{\tilde{R}_k}\right)_{*,Q} = \phi_{*,\phi(P_k)}^{-1} \circ (L_{R_k})_{*,\phi(Q)} \circ \phi_{*,Q}. \quad (\text{B.205})$$

Putting Eq. (B.205) into Eq. (B.201), we can obtain the result. $\square$

Now, we apply Thm. 195 to derive the expressions for our SPD MLRs presented in Thm. 117. In our cases of SPD MLRs, we set $Q = I$. For simplicity, we will omit the subscript k for P_k and A_k . We will first derive the expressions for SPD MLRs under (θ, α, β) -LEM, θ -LCM, (θ, α, β) -EM, and (θ, α, β) -AIM from Eq. (B.200). Then we will derive the expression for MLR under 2θ -BWM from Eq. (B.202). According to Thm. 187, the scaled metric ag shares the same Riemannian operators as g . We will use this fact throughout the following proof.

Proof. For simplicity, we abbreviate ϕ_θ as ϕ during the proof. Note that for 2θ -BWM, ϕ should be understood as $\phi_{2\theta}$. We first show $\phi(I)$ and the differential map $\phi_{*,I}$, which will be frequently required in the following proof:

$$\phi(I) = I, \quad (\text{B.206})$$

$$\phi_{*,I}(A) = \theta A, \forall A \in T_I \mathcal{S}_{++}^n. \quad (\text{B.207})$$

Let $\phi : (\mathcal{S}_{++}^n, \tilde{g}) \rightarrow (\mathcal{S}_{++}^n, g)$. Then the SPD MLR under $\tilde{g}$ by parallel transport with $Q = I$ is

$$p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[g_{\phi(P)}(\text{Log}_{\phi(P)} \phi(S), \Gamma_{I \rightarrow \phi(P)} \theta A) \right]. \quad (\text{B.208})$$

Next, we begin to prove the five SPD MLRs one by one.

(α, β) -LEM. As shown in Thm. 147, the standard LEM is the pullback metric from the Euclidean space $\mathcal{S}^n$. Similarly, (α, β) -LEM is also a pullback metric:

$$(\mathcal{S}_{++}^n, g^{(\alpha, \beta)\text{-LEM}}) \xrightarrow{\log} (\mathcal{S}^n, g^{(\alpha, \beta)}). \quad (\text{B.209})$$

By Eq. (B.200), we have

$$p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[\langle \log(S) - \log(P), \log_{*,I}(A) \rangle^{(\alpha, \beta)} \right] \quad (\text{B.210})$$

$$= \exp \left[\langle \log(S) - \log(P), A \rangle^{(\alpha, \beta)} \right]. \quad (\text{B.211})$$

θ -LCM. Simple computations show that θ -LCM is the scaled pullback metric of the

standard Euclidean metric in the Euclidean space of lower triangular matrices LT^n :

$$(\mathcal{S}_{++}^n, \theta^2 g^{\theta\text{-LCM}}) \xrightarrow{\phi} (\mathcal{S}_{++}^n, g^{\text{LCM}}) \xrightarrow{\text{Chol}} (\mathcal{L}_{++}^n, g^{\text{CM}}) \xrightarrow{\text{Dlog}} (\text{LT}^n, g^{\text{E}}), \quad (\text{B.212})$$

where g^{E} is the standard Frobenius inner product, and g^{CM} is the Cholesky metric on the Cholesky space $\mathcal{L}_{++}^n$ [137]. Denoting $\zeta = \text{Dlog} \circ \text{Chol} \circ \phi$, we have

$$\zeta_{*,I}(A) = \theta \left(\lfloor A \rfloor + \frac{1}{2} \mathbb{D}(A) \right), \forall A \in T_I \mathcal{S}_{++}^n. \quad (\text{B.213})$$

Similar to the case of (θ, α, β) -LEM, we have

$$p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[\frac{1}{\theta^2} \langle \zeta(S) - \zeta(P), \zeta_{*,I} A \rangle \right] \quad (\text{B.214})$$

$$= \exp \left[\frac{1}{\theta} \left\langle \begin{bmatrix} \lfloor \tilde{K} \rfloor - \lfloor \tilde{L} \rfloor + [\text{Dlog}(\mathbb{D}(\tilde{K})) - \text{Dlog}(\mathbb{D}(\tilde{L}))] \\ \lfloor A \rfloor + \frac{1}{2} \mathbb{D}(A) \end{bmatrix}, \begin{bmatrix} \tilde{K} \\ \tilde{L} \end{bmatrix} \right\rangle \right], \quad (\text{B.215})$$

where $\tilde{K} = \text{Chol}(S^\theta)$, $\tilde{L} = \text{Chol}(P^\theta)$, $\mathbb{D}(\tilde{K})$ is a diagonal matrix with diagonal elements from $\tilde{K}$, and $\lfloor \tilde{K} \rfloor$ is a strictly lower triangular matrix from $\tilde{K}$.

(θ, α, β) -EM. Let $\eta = \frac{1}{|\theta|} \phi$. A simple computation shows that (θ, α, β) -EM is the pullback metric of (α, β) -EM:

$$(\mathcal{S}_{++}^n, g^{(\theta, \alpha, \beta)\text{-EM}}) \xrightarrow{\eta} (\mathcal{S}_{++}^n, g^{(\alpha, \beta)\text{-EM}}). \quad (\text{B.216})$$

Besides, we have the following for η :

$$\eta_{*,I}(A) = \text{sgn}(\theta) A, \forall A \in T_I \mathcal{S}_{++}^n. \quad (\text{B.217})$$

According to Eq. (B.200), we have

$$p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp [\langle \eta(S) - \eta(P), \text{sgn}(\theta) A \rangle^{(\alpha, \beta)}] \quad (\text{B.218})$$

$$= \exp \left[\frac{1}{\theta} \langle S^\theta - P^\theta, A \rangle^{(\alpha, \beta)} \right]. \quad (\text{B.219})$$

(θ, α, β) -AIM. Putting $g^{(\alpha, \beta)\text{-AIM}}$ into Eq. (B.208), we have

$$p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[\frac{1}{\theta^2} g_{\phi(P)}^{(\alpha, \beta)\text{-AIM}} \left(P^{\frac{\theta}{2}} \log \left(P^{-\frac{\theta}{2}} S^{\theta} P^{-\frac{\theta}{2}} \right) P^{\frac{\theta}{2}}, P^{\frac{\theta}{2}} \theta A P^{\frac{\theta}{2}} \right) \right] \quad (\text{B.220})$$

$$= \exp \left[\frac{1}{\theta} \left\langle \log \left(P^{-\frac{\theta}{2}} S^{\theta} P^{-\frac{\theta}{2}} \right), A \right\rangle^{(\alpha, \beta)} \right]. \quad (\text{B.221})$$

2 θ -BWM. We first simplify Eq. (B.202) for SPD manifolds and then proceed to focus on the case of $g = g^{\text{BWM}}$. Denote $\phi : (\mathcal{S}_{++}^n, \tilde{g}, \tilde{\odot}) \rightarrow (\mathcal{S}_{++}^n, g, \odot)$, where the Lie group operation $\odot$ [195] is defined as

$$S_1 \odot S_2 = L_1 S_2 L_1^{\top}, \forall S_1, S_2 \in \mathcal{S}_{++}^n, \text{ with } L_1 = \text{Chol}(S_1). \quad (\text{B.222})$$

Note that I is the identity element of $(\mathcal{S}_{++}^n, \odot)$, and for any $S \in \mathcal{S}_{++}^n$, the differential map of the left translation L_S under $\odot$ is

$$(L_S)_{*,Q}(V) = L V L^{\top}, \forall Q \in \mathcal{S}_{++}^n, \forall V \in T_Q \mathcal{S}_{++}^n, \text{ with } L = \text{Chol}(S). \quad (\text{B.223})$$

For the induced Lie group $(\mathcal{S}_{++}^n, \tilde{\odot})$, the left translation $\tilde{L}_{P \tilde{\odot} I_{\tilde{\odot}}^{-1}}$ under $\tilde{\odot}$ is

$$\tilde{L}_{P \tilde{\odot} I_{\tilde{\odot}}^{-1}} = \phi^{-1} \circ L_{\phi(P) \odot \phi(I)_{\odot}^{-1}} \circ \phi, \quad (\text{B.224})$$

$$= \phi^{-1} \circ L_{P^{2\theta}} \circ \phi, \quad \phi(P) \odot \phi(I)_{\odot}^{-1} = P^{2\theta}. \quad (\text{B.225})$$

The associated differential at I is

$$\left(\tilde{L}_{P \tilde{\odot} I_{\tilde{\odot}}^{-1}} \right)_{*,I}(A) = \phi_{*,\phi(P)}^{-1} \circ (L_{P^{2\theta}})_{*,\phi(I)} \circ \phi_{*,I}(A) \quad (\text{B.226})$$

$$= 2\theta \phi_{*,\phi(P)}^{-1}(\bar{L} A \bar{L}^{\top}), \quad (\text{B.227})$$

where $\bar{L} = \text{Chol}(P^{2\theta})$. Then the SPD MLR under $\tilde{g}$ and $\tilde{\odot}$ by left translation is

$$p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[2\theta g_{\phi(P)} \left(\text{Log}_{\phi(P)} \phi(S), \bar{L} A \bar{L}^{\top} \right) \right]. \quad (\text{B.228})$$

Setting $g = g^{\text{BWM}}$ (we omit the scaling factor), we obtain the SPD MLR under 2 θ -BWM:

$$p(y = k \mid S \in \mathcal{S}_{++}^n) \propto \exp \left[2\theta \cdot \frac{1}{4\theta^2} g_{\phi(P)}^{\text{BWM}} \left(\text{Log}_{\phi(P)}^{\text{BWM}} \phi(S), \bar{L} A \bar{L}^{\top} \right) \right] \quad (\text{B.229})$$

$$= \exp \left[\frac{1}{4\theta} \langle (P^{2\theta} S^{2\theta})^{\frac{1}{2}} + (S^{2\theta} P^{2\theta})^{\frac{1}{2}} - 2P^{2\theta}, \mathcal{L}_{P^{2\theta}}[\bar{L}A\bar{L}^\top] \rangle \right]. \quad (\text{B.230})$$

□

B.5.5 Proof of Thm. 119

Proof. During this proof, we use the ambient representation of tangent vectors. Given rotation matrices $P, Q \in \text{SO}(n)$ and a tangent vector $H \in T_Q \text{SO}(n)$, let $c(t)$ be a curve on $\text{SO}(n)$ satisfying $c(0) = Q$ and $c'(0) = H$. The differential of the left translation $L_{PQ^{-1}}$ at Q is

$$(L_{PQ^{-1}})_{*,Q}(H) = \left. \frac{dPQ^{-1}c(t)}{dt} \right|_{t=0} = PQ^{-1}H = PQ^\top H, \quad (\text{B.231})$$

which is precisely the vector transport $\mathcal{T}_{Q \rightarrow P}(H)$ in Boumal and Absil [28, Tab. 1]. □

B.5.6 Proof of Thm. 120

Proof. Setting $Q = I$ in Eq. (4.28) and using Thm. 119 give $\tilde{A}_k = P_k A_k$. Combining this identity with the logarithmic map and metric in Tab. 2.12, we obtain

$$\left\langle \text{Log}_{P_k} R, \tilde{A}_k \right\rangle_{P_k} = \langle P_k \log(P_k^\top R), P_k A_k \rangle = \langle \log(P_k^\top R), A_k \rangle. \quad (\text{B.232})$$

Substituting this expression into Eq. (4.26) yields the result. □

B.6 Proper Velocity Neural Networks

B.6.1 Derivation of the Proper Velocity Metric

The PV line element at $x \in \mathbb{PV}_K^n$ can be written in terms of the curvature parameter $K < 0$ as

$$Q_x(u) = \|u\|^2 + K\beta_x^2 \langle x, u \rangle^2, \quad \forall u \in T_x \mathbb{PV}_K^n \simeq \mathbb{R}^n, \quad (\text{B.233})$$

where $\beta_x = \frac{1}{\sqrt{1-K\|x\|^2}}$. This is equivalent to the expression in Ungar [200, Eq. (7.76)] after substituting $s^2 = -1/K$. Given $u, v \in T_x \mathbb{PV}_K^n$, the bilinear form $g_x(u, v)$ is

obtained by the polarization identity:

$$g_x(u, v) = \frac{1}{4} (Q_x(u + v) - Q_x(u - v)). \quad (\text{B.234})$$

We first expand the two terms in the polarization identity:

$$\begin{aligned} Q_x(u + v) &= \|u + v\|^2 + K\beta_x^2 \langle x, u + v \rangle^2 \\ &= \|u\|^2 + 2\langle u, v \rangle + \|v\|^2 + K\beta_x^2 (\langle x, u \rangle^2 + 2\langle x, u \rangle \langle x, v \rangle + \langle x, v \rangle^2), \\ Q_x(u - v) &= \|u - v\|^2 + K\beta_x^2 \langle x, u - v \rangle^2 \\ &= \|u\|^2 - 2\langle u, v \rangle + \|v\|^2 + K\beta_x^2 (\langle x, u \rangle^2 - 2\langle x, u \rangle \langle x, v \rangle + \langle x, v \rangle^2). \end{aligned} \quad (\text{B.235})$$

Taking the difference yields

$$Q_x(u + v) - Q_x(u - v) = 4\langle u, v \rangle + 4K\beta_x^2 \langle x, u \rangle \langle x, v \rangle. \quad (\text{B.236})$$

Substituting this expression into the polarization identity, we obtain

$$\begin{aligned} g_x(u, v) &= \frac{1}{4} (Q_x(u + v) - Q_x(u - v)) \\ &= \langle u, v \rangle + K\beta_x^2 \langle x, u \rangle \langle x, v \rangle, \end{aligned} \quad (\text{B.237})$$

which coincides with the expression of the PV metric in Eq. (5.1).

B.6.2 Proof of Thm. 121

Proof. Differential of $\pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}$. Consider the curve $c : (-\varepsilon, \varepsilon) \rightarrow \mathbb{P}\mathbb{V}_K^n$ which satisfies $c(0) = x$ and $c'(0) = v$. By definition of the differential,

$$d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v) = \left. \frac{d}{dt} \right|_{t=0} \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(c(t)). \quad (\text{B.238})$$

Using $\pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(x) = \frac{\beta_x}{1+\beta_x}x$ with $\beta_x = \frac{1}{\sqrt{1-K\|x\|^2}}$, we write

$$\pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(c(t)) = h(t)c(t), \quad h(t) := \frac{\beta_{c(t)}}{1 + \beta_{c(t)}}. \quad (\text{B.239})$$

Let $r(t) = \|c(t)\|^2$, so that $\beta_{c(t)} = (1 - Kr(t))^{-1/2}$. Then

$$r'(0) = 2\langle x, v \rangle, \quad \beta'_{c(0)} = \frac{1}{2}(1 - Kr(0))^{-3/2}Kr'(0) = K\beta_x^3 \langle x, v \rangle. \quad (\text{B.240})$$

Differentiating $h(t)$ at $t = 0$ gives

$$h'(0) = \frac{\beta'_{c(0)}}{(1 + \beta_x)^2} = K \frac{\beta_x^3}{(1 + \beta_x)^2} \langle x, v \rangle. \quad (\text{B.241})$$

Finally, differentiating $h(t)c(t)$ at $t = 0$ yields

$$\begin{aligned} d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v) &= h'(0)x + h(0)v \\ &= K \frac{\beta_x^3}{(1 + \beta_x)^2} \langle x, v \rangle x + \frac{\beta_x}{1 + \beta_x} v. \end{aligned} \quad (\text{B.242})$$

In particular, at $x = \mathbf{0}$ one has $\beta_{\mathbf{0}} = 1$ and $\langle x, v \rangle = 0$. Thus, we have

$$d_{\mathbf{0}} \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v) = \frac{\beta_{\mathbf{0}}}{1 + \beta_{\mathbf{0}}} v = \frac{1}{2} v. \quad (\text{B.243})$$

Differential of $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}$. Consider the curve $c : (-\varepsilon, \varepsilon) \rightarrow \mathbb{P}_K^n$ which satisfies $c(0) = y$ and $c'(0) = w$. By definition of the differential,

$$d_y \pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(w) = \left. \frac{d}{dt} \right|_{t=0} \pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(c(t)). \quad (\text{B.244})$$

Using the explicit expression $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(y) = 2\gamma_y^2 y$ with $\gamma_y = \frac{1}{\sqrt{1+K\|y\|^2}}$, we obtain

$$\pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(c(t)) = 2\gamma_{c(t)}^2 c(t). \quad (\text{B.245})$$

Let $r(t) = \|c(t)\|^2$ so that $\gamma_{c(t)}^2 = (1 + Kr(t))^{-1}$. Then

$$r'(0) = 2 \langle y, w \rangle, \quad \left. \frac{d}{dt} \right|_{t=0} \gamma_{c(t)}^2 = -\frac{Kr'(0)}{(1 + Kr(0))^2} = -2K\gamma_y^4 \langle y, w \rangle. \quad (\text{B.246})$$

Differentiating $2\gamma_{c(t)}^2 c(t)$ at $t = 0$ yields

$$\begin{aligned} d_y \pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(w) &= 2 \left. \frac{d}{dt} \right|_{t=0} \gamma_{c(t)}^2 y + 2\gamma_y^2 w \\ &= -4K\gamma_y^4 \langle y, w \rangle y + 2\gamma_y^2 w. \end{aligned} \quad (\text{B.247})$$

In particular, at $y = \mathbf{0}$ we have $\gamma_{\mathbf{0}} = 1$ and $\langle y, w \rangle = 0$. Thus, we have

$$d_{\mathbf{0}} \pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(w) = 2w. \quad (\text{B.248})$$

□

B.6.3 Proof of Thm. 122

Proof. It suffices to show that for any $x \in \mathbb{P}\mathbb{V}_K^n$ and $v, w \in T_x \mathbb{P}\mathbb{V}_K^n$,

$$g_y^{\mathbb{P}}(d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v), d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(w)) = g_x^{\mathbb{P}\mathbb{V}}(v, w), \quad (\text{B.249})$$

where $y = \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(x)$.

We first recall the following equations from Eq. (5.1), Sec. 2.9.5, and Thm. 121:

$$\begin{aligned} g_x^{\mathbb{P}\mathbb{V}}(v, w) &= \langle v, w \rangle + K \beta_x^2 \langle x, v \rangle \langle x, w \rangle, \quad \forall x \in \mathbb{P}\mathbb{V}_K^n, \forall v, w \in T_x \mathbb{P}\mathbb{V}_K^n, \\ g_y^{\mathbb{P}}(u, z) &= (\lambda_y^K)^2 \langle u, z \rangle, \quad \forall y \in \mathbb{P}_K^n, \forall u, z \in T_y \mathbb{P}_K^n, \\ \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(x) &= \frac{\beta_x}{1 + \beta_x} x, \quad \forall x \in \mathbb{P}\mathbb{V}_K^n, \\ d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v) &= \frac{\beta_x}{1 + \beta_x} v + K \frac{\beta_x^3}{(1 + \beta_x)^2} \langle x, v \rangle x, \quad \forall x \in \mathbb{P}\mathbb{V}_K^n, \forall v \in T_x \mathbb{P}\mathbb{V}_K^n. \end{aligned} \quad (\text{B.250})$$

Let

$$a = \frac{\beta_x}{1 + \beta_x}, \quad b = K \frac{\beta_x^3}{(1 + \beta_x)^2}. \quad (\text{B.251})$$

Then $d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v) = av + b \langle x, v \rangle x$ and $d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(w) = aw + b \langle x, w \rangle x$. Thus,

$$\begin{aligned} &g_y^{\mathbb{P}}(d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v), d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(w)) \\ &= (\lambda_y^K)^2 \langle av + b \langle x, v \rangle x, aw + b \langle x, w \rangle x \rangle \\ &= (\lambda_y^K)^2 (a^2 \langle v, w \rangle + ab \langle x, w \rangle \langle v, x \rangle + ab \langle x, v \rangle \langle x, w \rangle + b^2 \langle x, v \rangle \langle x, w \rangle \langle x, x \rangle) \\ &= (\lambda_y^K)^2 (a^2 \langle v, w \rangle + (2ab + b^2 \|x\|^2) \langle x, v \rangle \langle x, w \rangle). \end{aligned} \quad (\text{B.252})$$

Using $y = \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(x)$ and the relation between λ_y^K , β_x , and $\|x\|$ from Sec. 2.9.5, we simplify the coefficients. First,

$$\begin{aligned} K \|y\|^2 &= K \left\langle \frac{\beta_x}{1 + \beta_x} x, \frac{\beta_x}{1 + \beta_x} x \right\rangle \\ &= K \frac{\beta_x^2}{(1 + \beta_x)^2} \|x\|^2 \\ &= \frac{\beta_x^2 - 1}{(1 + \beta_x)^2}, \end{aligned} \quad (\text{B.253})$$

where we use $\beta_x^2 = \frac{1}{1-K\|x\|^2}$. Hence

$$1 + K\|y\|^2 = 1 + \frac{\beta_x^2 - 1}{(1 + \beta_x)^2} = \frac{2\beta_x}{1 + \beta_x}, \quad (\text{B.254})$$

which implies $\lambda_y^K = \frac{2}{1+K\|y\|^2} = \frac{1+\beta_x}{\beta_x}$. Therefore

$$(\lambda_y^K)^2 a^2 = \left(\frac{1 + \beta_x}{\beta_x} \right)^2 \left(\frac{\beta_x}{1 + \beta_x} \right)^2 = 1. \quad (\text{B.255})$$

Next, we compute

$$\begin{aligned} 2ab &= 2 \frac{\beta_x}{1 + \beta_x} K \frac{\beta_x^3}{(1 + \beta_x)^2} = \frac{2K\beta_x^4}{(1 + \beta_x)^3}, \\ b^2 \|x\|^2 &= K^2 \frac{\beta_x^6}{(1 + \beta_x)^4} \|x\|^2 = K \frac{\beta_x^4(\beta_x^2 - 1)}{(1 + \beta_x)^4}, \end{aligned} \quad (\text{B.256})$$

which brings us to

$$\begin{aligned} 2ab + b^2 \|x\|^2 &= K \frac{\beta_x^4}{(1 + \beta_x)^4} (2(1 + \beta_x) + \beta_x^2 - 1) \\ &= K \frac{\beta_x^4(\beta_x + 1)^2}{(1 + \beta_x)^4} = K \frac{\beta_x^4}{(1 + \beta_x)^2}. \end{aligned} \quad (\text{B.257})$$

Multiplying by $(\lambda_y^K)^2 = \left(\frac{1+\beta_x}{\beta_x} \right)^2$ yields

$$(\lambda_y^K)^2 (2ab + b^2 \|x\|^2) = \left(\frac{1 + \beta_x}{\beta_x} \right)^2 K \frac{\beta_x^4}{(1 + \beta_x)^2} = K\beta_x^2. \quad (\text{B.258})$$

Substituting these identities into the expression for $g_{\mathbb{P}}$ gives

$$g_y^{\mathbb{P}} (d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(v), d_x \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(w)) = \langle v, w \rangle + K\beta_x^2 \langle x, v \rangle \langle x, w \rangle = g_x^{\mathbb{P}\mathbb{V}}(v, w). \quad (\text{B.259})$$

□

B.6.4 Proof of Thm. 123

Following the notation in the main theorem, we further denote:

$$\bar{x} = \pi(x) \in \mathbb{P}_K^n, \quad \bar{y} = \pi(y) \in \mathbb{P}_K^n, \quad \bar{v} = d_x \pi(v) \in T_{\bar{x}} \mathbb{P}_K^n, \quad (\text{B.260})$$

Recalling Eq. (5.8) and Thm. 121, we have the following:

$$\pi(x) = \frac{\beta_x}{1 + \beta_x} x, \quad \forall x \in \mathbb{P}\mathbb{V}_K^n, \quad (\text{B.261})$$

$$\pi^{-1}(\bar{y}) = 2\gamma_{\bar{y}}^2 \bar{y}, \quad \forall \bar{y} \in \mathbb{P}_K^n, \quad (\text{B.262})$$

$$d_x \pi(v) = K \frac{\beta_x^3}{(1 + \beta_x)^2} \langle x, v \rangle x + \frac{\beta_x}{1 + \beta_x} v, \quad \forall x \in \mathbb{P}\mathbb{V}_K^n, \forall v \in T_x \mathbb{P}\mathbb{V}_K^n, \quad (\text{B.263})$$

$$d_{\bar{y}} \pi^{-1}(w) = -4K\gamma_{\bar{y}}^4 \langle \bar{y}, w \rangle \bar{y} + 2\gamma_{\bar{y}}^2 w, \quad \forall \bar{y} \in \mathbb{P}_K^n, \forall w \in T_{\bar{y}} \mathbb{P}_K^n. \quad (\text{B.264})$$

Next, we derive the expressions for each PV operator.

B.6.4.1 PV Exponential Map

We recall from Tab. 2.14 that the Riemannian exponential on the Poincaré ball is

$$\text{Exp}_x^{\mathbb{P}}(\bar{v}) = \bar{x} \oplus_{\text{M}} \left(\frac{1}{\sqrt{-K}} \tanh \left(\frac{\sqrt{-K} \lambda_x^K \|\bar{v}\|}{2} \right) \frac{\bar{v}}{\|\bar{v}\|} \right). \quad (\text{B.265})$$

By the Riemannian isometry and the gyrovector isomorphism of π , for any $x \in \mathbb{P}\mathbb{V}_K^n$ and $v \in T_x \mathbb{P}\mathbb{V}_K^n$ we have

$$\begin{aligned} \text{Exp}_x(v) &\stackrel{(1)}{=} \pi^{-1}(\text{Exp}_x^{\mathbb{P}}(\bar{v})) \\ &\stackrel{(2)}{=} x \oplus_{\text{U}} \pi^{-1} \left(\frac{1}{\sqrt{-K}} \tanh \left(\frac{\sqrt{-K} \lambda_x^K \|\bar{v}\|}{2} \right) \frac{\bar{v}}{\|\bar{v}\|} \right), \end{aligned} \quad (\text{B.266})$$

The above equalities follow from the following facts.

(1) Isometry.

(2) Gyrovector isomorphism.

Let

$$u = \frac{1}{\sqrt{-K}} \tanh \left(\frac{\sqrt{-K} \lambda_x^K \|\bar{v}\|}{2} \right) \frac{\bar{v}}{\|\bar{v}\|}. \quad (\text{B.267})$$

We have

$$\|u\| = \frac{1}{\sqrt{-K}} \tanh \left(\frac{\sqrt{-K} \lambda_x^K \|\bar{v}\|}{2} \right). \quad (\text{B.268})$$

Let $t = \frac{\sqrt{-K}\lambda_{\bar{x}}^K\|\bar{v}\|}{2}$ so that $\sqrt{-K}\|u\| = \tanh(t)$. Then

$$\begin{aligned}
 \pi^{-1}(u) &= \frac{2}{1 + K\|u\|^2}u \\
 &= \frac{2}{1 + K\|u\|^2} \left(\frac{1}{\sqrt{-K}} \tanh(t) \frac{\bar{v}}{\|\bar{v}\|} \right) \\
 &= \frac{2}{1 - \tanh^2(t)} \frac{1}{\sqrt{-K}} \tanh(t) \frac{\bar{v}}{\|\bar{v}\|} \\
 &= \frac{2}{\sqrt{-K}} \frac{\tanh(t)}{1 - \tanh^2(t)} \frac{\bar{v}}{\|\bar{v}\|} \\
 &\stackrel{(1)}{=} \frac{2}{\sqrt{-K}} \tanh(t) \cosh^2(t) \frac{\bar{v}}{\|\bar{v}\|} \\
 &= \frac{2}{\sqrt{-K}} \frac{\sinh(t)}{\cosh(t)} \cosh^2(t) \frac{\bar{v}}{\|\bar{v}\|} \\
 &= \frac{2}{\sqrt{-K}} \sinh(t) \cosh(t) \frac{\bar{v}}{\|\bar{v}\|} \\
 &\stackrel{(2)}{=} \frac{1}{\sqrt{-K}} (2 \sinh(t) \cosh(t)) \frac{\bar{v}}{\|\bar{v}\|} \\
 &= \frac{1}{\sqrt{-K}} \sinh(2t) \frac{\bar{v}}{\|\bar{v}\|} \\
 &= \frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} \lambda_{\bar{x}}^K \|\bar{v}\| \right) \frac{\bar{v}}{\|\bar{v}\|} \\
 &\stackrel{(3)}{=} \frac{1}{\sqrt{-K}} \sinh \left(\frac{\sqrt{-K}(1 + \beta_x)}{\beta_x} \|\bar{v}\| \right) \frac{\bar{v}}{\|\bar{v}\|}.
 \end{aligned} \tag{B.269}$$

The above equalities use:

$$(1) \quad 1 - \tanh^2(t) = 1/\cosh^2(t);$$

$$(2) \quad \sinh(2t) = 2 \sinh(t) \cosh(t);$$

$$(3) \quad \lambda_{\bar{x}}^K = \frac{1 + \beta_x}{\beta_x}.$$

B.6.4.2 PV Logarithmic Map

We recall from Tab. 2.14 that the Riemannian logarithm on the Poincaré ball is

$$\text{Log}_{\mathbb{P}}^{\mathbb{P}}(\bar{y}) = \frac{2}{\sqrt{-K}\lambda_{\bar{x}}^K} \frac{\tanh^{-1}(\sqrt{-K}\|\bar{z}\|)}{\|\bar{z}\|} \bar{z}, \quad \bar{z} = (-\bar{x}) \oplus_{\mathbf{M}} \bar{y}, \tag{B.270}$$

where $\lambda_{\bar{x}}^K = \frac{2}{1+K\|\bar{x}\|^2}$. We define

$$z = (-x) \oplus_{\mathbb{U}} y, \quad \bar{z} = \pi(z). \quad (\text{B.271})$$

By the Riemannian isometry of π , we have

$$\begin{aligned} \text{Log}_x(y) &= d_{\bar{x}}\pi^{-1}(\text{Log}_{\bar{x}}^{\mathbb{P}}(\bar{y})) \\ &= d_{\bar{x}}\pi^{-1}\left(\frac{2}{\sqrt{-K}\lambda_{\bar{x}}^K} \frac{\tanh^{-1}(\sqrt{-K}\|\bar{z}\|)}{\|\bar{z}\|} \bar{z}\right) \\ &= \alpha(x, y) d_{\bar{x}}\pi^{-1}(\bar{z}), \end{aligned} \quad (\text{B.272})$$

where

$$\alpha(x, y) = \frac{2}{\sqrt{-K}\lambda_{\bar{x}}^K} \frac{\tanh^{-1}(\sqrt{-K}\|\bar{z}\|)}{\|\bar{z}\|}. \quad (\text{B.273})$$

The differential of π^{-1} at $\bar{x} = \pi(x)$ is

$$d_{\bar{x}}\pi^{-1}(h) = -4K\gamma_{\bar{x}}^4 \langle \bar{x}, h \rangle \bar{x} + 2\gamma_{\bar{x}}^2 h, \quad \forall h \in T_{\bar{x}}\mathbb{P}_K^n, \quad (\text{B.274})$$

where $\gamma_{\bar{x}} = \frac{1}{\sqrt{1+K\|\bar{x}\|^2}}$. Using $\bar{x} = \frac{\beta_x}{1+\beta_x}x$ and the relation $1 - K\|x\|^2 = \frac{1}{\beta_x^2}$, we have

$$\begin{aligned} K\|\bar{x}\|^2 &= K \frac{\beta_x^2}{(1+\beta_x)^2} \|x\|^2 = \frac{\beta_x^2}{(1+\beta_x)^2} \left(\frac{\beta_x^2 - 1}{\beta_x^2} \right) = \frac{\beta_x^2 - 1}{(1+\beta_x)^2}, \\ 1 + K\|\bar{x}\|^2 &= 1 + \frac{\beta_x^2 - 1}{(1+\beta_x)^2} = \frac{2\beta_x}{1+\beta_x}. \end{aligned} \quad (\text{B.275})$$

Hence

$$\gamma_{\bar{x}}^2 = \frac{1}{1 + K\|\bar{x}\|^2} = \frac{1 + \beta_x}{2\beta_x}, \quad \gamma_{\bar{x}}^4 = \left(\frac{1 + \beta_x}{2\beta_x} \right)^2 = \frac{(1 + \beta_x)^2}{4\beta_x^2}. \quad (\text{B.276})$$

Substituting these into $d_{\bar{x}}\pi^{-1}(h)$ yields

$$\begin{aligned} d_{\bar{x}}\pi^{-1}(h) &= -4K \frac{(1 + \beta_x)^2}{4\beta_x^2} \langle \bar{x}, h \rangle \bar{x} + 2 \frac{1 + \beta_x}{2\beta_x} h \\ &= -K \frac{(1 + \beta_x)^2}{\beta_x^2} \langle \bar{x}, h \rangle \bar{x} + \frac{1 + \beta_x}{\beta_x} h \\ &= \frac{1 + \beta_x}{\beta_x} h - K \langle x, h \rangle x, \end{aligned} \quad (\text{B.277})$$

where the last equality uses that $\bar{x} = \frac{\beta_x}{1+\beta_x}x$. Applying Eq. (B.277) with $h = \bar{z}$ and

using that $\bar{z} = \pi(z)$ is collinear with $z = (-x) \oplus_{\mathbb{U}} y$, we obtain

$$\begin{aligned} \text{Log}_x(y) &= \alpha(x, y)(d_x \pi)^{-1}(\bar{z}) \\ &= \alpha(x, y) \left(\frac{1 + \beta_x}{\beta_x} \bar{z} - K \langle x, \bar{z} \rangle x \right). \end{aligned} \quad (\text{B.278})$$

Since $\bar{z} = \pi(z)$ and π is given by Eq. (5.8), z and $\bar{z}$ are collinear and

$$\bar{z} = \rho z, \quad \rho = \frac{\beta_z}{1 + \beta_z}, \quad (\text{B.279})$$

which also implies $\langle x, \bar{z} \rangle = \rho \langle x, z \rangle$. Substituting these into Eq. (B.278) yields

$$\begin{aligned} \text{Log}_x(y) &= \alpha(x, y) \left(\frac{1 + \beta_x}{\beta_x} \rho z - K \rho \langle x, z \rangle x \right) \\ &= \underbrace{\alpha(x, y) \frac{1 + \beta_x}{\beta_x} \rho z}_{\sigma(x, y)} + \underbrace{(-K \alpha(x, y) \rho) \langle x, z \rangle x}_{\tau(x, y)}. \end{aligned} \quad (\text{B.280})$$

Using the definition of $\alpha(x, y)$ in Eq. (B.273) together with $\lambda_{\bar{x}}^K = \frac{1 + \beta_x}{\beta_x}$ and $\rho = \frac{\beta_z}{1 + \beta_z}$, a straightforward simplification yields

$$\sigma(x, y) = \frac{2}{\sqrt{-K}} \frac{\tanh^{-1}(\sqrt{-K} \|\bar{z}\|)}{\|z\|}, \quad \tau(x, y) = \frac{2\beta_x}{1 + \beta_x} \frac{\sqrt{-K} \tanh^{-1}(\sqrt{-K} \|\bar{z}\|)}{\|z\|}. \quad (\text{B.281})$$

Thus,

$$\text{Log}_x(y) = \sigma(x, y)z + \tau(x, y) \langle x, z \rangle x. \quad (\text{B.282})$$

B.6.4.3 PV Parallel Transport

We recall from Tab. 2.14 that the parallel transport on the Poincaré ball is

$$\text{PT}_{\bar{x} \rightarrow \bar{y}}^{\mathbb{P}}(w) = \frac{\lambda_{\bar{x}}^K}{\lambda_{\bar{y}}^K} \text{gyr}_{\mathbb{M}}[\bar{y}, -\bar{x}](w), \quad \text{with } w \in T_{\bar{x}} \mathbb{P}_K^n. \quad (\text{B.283})$$

We have

$$\begin{aligned}
 & \text{PT}_{x \rightarrow y}(v) \\
 & \stackrel{(1)}{=} d_{\bar{y}} \pi^{-1} \left(\text{PT}_{\bar{x} \rightarrow \bar{y}}^{\mathbb{P}}(d_x \pi(v)) \right) \\
 & = d_{\bar{y}} \pi^{-1} \left(\frac{\lambda_{\bar{x}}^K}{\lambda_{\bar{y}}^K} \text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) \right) \\
 & \stackrel{(2)}{=} \frac{\lambda_{\bar{x}}^K}{\lambda_{\bar{y}}^K} d_{\bar{y}} \pi^{-1} \left(\text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) \right) \\
 & \stackrel{(3)}{=} \frac{\lambda_{\bar{x}}^K}{\lambda_{\bar{y}}^K} \left(\frac{1 + \beta_y}{\beta_y} \text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) - K \langle y, \text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) \rangle y \right) \\
 & \stackrel{(4)}{=} \frac{(1 + \beta_x) \beta_y}{(1 + \beta_y) \beta_x} \left(\frac{1 + \beta_y}{\beta_y} \text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) - K \langle y, \text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) \rangle y \right) \\
 & = \frac{1 + \beta_x}{\beta_x} \text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) - K \frac{(1 + \beta_x) \beta_y}{(1 + \beta_y) \beta_x} \langle y, \text{gyr}_{\text{M}}[\bar{y}, -\bar{x}](d_x \pi(v)) \rangle y.
 \end{aligned} \tag{B.284}$$

The above equalities use:

- (1) the isometry property of π ;
- (2) linearity of $d_{\bar{y}} \pi^{-1}$;
- (3) Eq. (B.277).
- (4) Using the relation between $\lambda_{\bar{x}}^K$ and β_x in the proof of Thm. 122,

$$\lambda_{\bar{x}}^K = \frac{1 + \beta_x}{\beta_x} \quad \lambda_{\bar{y}}^K = \frac{1 + \beta_y}{\beta_y}. \tag{B.285}$$

B.6.4.4 PV Geodesic Distance

We recall from Tab. 2.14 that the geodesic distance on the Poincaré ball $\mathbb{P}_K^n$ is

$$d^{\mathbb{P}}(y_1, y_2) = \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \|(-y_1) \oplus_{\text{M}} y_2\| \right), \quad y_1, y_2 \in \mathbb{P}_K^n. \tag{B.286}$$

By isometry and isomorphism, the PV geodesic distance is

$$\begin{aligned}
 d(x, y) &= d^{\mathbb{P}}(\pi(x), \pi(y)) \\
 &= \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \|(-\pi(x)) \oplus_{\text{M}} \pi(y)\| \right) \\
 &= \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \|\pi(-x \oplus_{\text{U}} y)\| \right).
 \end{aligned} \tag{B.287}$$

B.6.4.5 Special Cases at the Identity

Exponential Map at the Identity.

$$\begin{aligned} \text{Exp}_{\mathbf{0}}(v) &\stackrel{(1)}{=} \frac{1}{\sqrt{-K}} \sinh \left(\frac{\sqrt{-K}(1 + \beta_{\mathbf{0}})}{\beta_{\mathbf{0}}} \|\bar{v}\| \right) \frac{\bar{v}}{\|\bar{v}\|} \\ &\stackrel{(2)}{=} \frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} \|v\| \right) \frac{v}{\|v\|}. \end{aligned} \tag{B.288}$$

The above comes from the following.

(1) $\mathbf{0}$ is the gyro identity;

(2) $\beta_{\mathbf{0}} = 1$ and $d_{\mathbf{0}}\pi(v) = \frac{1}{2}v$.

Logarithmic Map at the Identity. As $z = -\mathbf{0} \oplus_{\mathbf{U}} y = y$, we have

$$\begin{aligned} \text{Log}_{\mathbf{0}}(y) &= \sigma(\mathbf{0}, y)z + \tau(\mathbf{0}, y) \langle \mathbf{0}, z \rangle \mathbf{0} \\ &= \sigma(\mathbf{0}, y)y \\ &= \frac{2}{\sqrt{-K}} \frac{\tanh^{-1}(\sqrt{-K} \|\pi(y)\|)}{\|y\|} y. \end{aligned} \tag{B.289}$$

From $\pi(y) = \frac{\beta_y}{1+\beta_y}y$ and $\beta_y = \frac{1}{\sqrt{1-K\|y\|^2}}$, we obtain

$$\sqrt{-K} \|\pi(y)\| = \frac{\beta_y}{1 + \beta_y} \sqrt{-K} \|y\|. \tag{B.290}$$

Let $t = \sqrt{-K} \|y\|$ and $s = \sqrt{1+t^2}$, so that $\beta_y = \frac{1}{\sqrt{1-K\|y\|^2}} = \frac{1}{s}$. Define

$$a = \frac{\beta_y}{1 + \beta_y} t = \frac{t}{s + 1}. \tag{B.291}$$

Using the hyperbolic double-angle identity, we have

$$\begin{aligned}
 \tanh(2 \tanh^{-1}(a)) &= \frac{2a}{1+a^2} \\
 &= \frac{2t/(s+1)}{1+t^2/(s+1)^2} \\
 &= \frac{2t(s+1)}{(s+1)^2+t^2} \\
 &= \frac{2t(s+1)}{s^2+2s+1+t^2} \\
 &= \frac{2t(s+1)}{2(1+t^2+s)} \\
 &= \frac{t(s+1)}{1+t^2+s} \\
 &= \frac{t(s+1)}{s^2+s} \\
 &= \frac{t}{s} = \frac{t}{\sqrt{1+t^2}}.
 \end{aligned} \tag{B.292}$$

Denoting $u = \sinh^{-1}(t)$, we have

$$\cosh(u) = \sqrt{1 + \sinh^2(u)} = \sqrt{1 + t^2}. \tag{B.293}$$

Therefore,

$$\tanh(\sinh^{-1}(t)) = \tanh(u) = \frac{\sinh(u)}{\cosh(u)} = \frac{t}{\sqrt{1+t^2}}. \tag{B.294}$$

Since $\tanh$ is strictly increasing on $\mathbb{R}$, this implies that

$$2 \tanh^{-1}\left(\frac{\beta_y}{1+\beta_y}t\right) = \sinh^{-1}(t). \tag{B.295}$$

Substituting this identity back gives

$$\sigma(\mathbf{0}, y) = \frac{1}{\sqrt{-K}} \frac{\sinh^{-1}(\sqrt{-K} \|y\|)}{\|y\|}, \tag{B.296}$$

and therefore

$$\text{Log}_0(y) = \frac{1}{\sqrt{-K}} \sinh^{-1}(\sqrt{-K} \|y\|) \frac{y}{\|y\|}. \tag{B.297}$$

Parallel Transport from the Identity. The gyration satisfies

$$\text{gyr}_M[\mathbf{0}, \bar{y}] = \text{gyr}_M[\bar{y}, \mathbf{0}] = \text{id}. \tag{B.298}$$

Substituting this into Eq. (B.284) gives

$$\begin{aligned}
 \text{PT}_{\mathbf{0} \rightarrow y}(v) &= \frac{1 + \beta_{\mathbf{0}}}{\beta_{\mathbf{0}}} d_{\mathbf{0}} \pi(v) - K \frac{(1 + \beta_{\mathbf{0}}) \beta_y}{(1 + \beta_y) \beta_{\mathbf{0}}} \langle y, d_{\mathbf{0}} \pi(v) \rangle y \\
 &= 2 \cdot \frac{1}{2} v - K \frac{2\beta_y}{1 + \beta_y} \cdot \frac{1}{2} \langle y, v \rangle y \\
 &= v - K \frac{\beta_y}{1 + \beta_y} \langle y, v \rangle y.
 \end{aligned} \tag{B.299}$$

Parallel Transport to the Identity. Taking $y = \mathbf{0}$ in Eq. (B.284) and using $\text{gyr}_M[\mathbf{0}, -\bar{x}] = \text{id}$ yields

$$\begin{aligned}
 \text{PT}_{x \rightarrow \mathbf{0}}(v) &= \frac{1 + \beta_x}{\beta_x} d_x \pi(v) \\
 &= \frac{1 + \beta_x}{\beta_x} \left(K \frac{\beta_x^3}{(1 + \beta_x)^2} \langle x, v \rangle x + \frac{\beta_x}{1 + \beta_x} v \right) \\
 &= v + K \frac{\beta_x^2}{1 + \beta_x} \langle x, v \rangle x.
 \end{aligned} \tag{B.300}$$

Distance from the Identity. This can be directly obtained by the gyro identity.

$$d(\mathbf{0}, y) = \frac{2}{\sqrt{-K}} \tanh^{-1} \left(\sqrt{-K} \|\pi(y)\| \right). \tag{B.301}$$

Using the same identity as above with $t = \sqrt{-K} \|y\|$ yields

$$d(\mathbf{0}, y) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\sqrt{-K} \|y\| \right). \tag{B.302}$$

B.6.5 Proof of Thm. 124

Proof. As shown in Thm. 88, the Möbius gyroaddition and gyromultiplication can be written by the Riemannian operators. Besides, the isometry $\pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}$ preserves the identity: $\pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}(\mathbf{0}) = \mathbf{0}$. By Nguyen and Yang [159, Lems. 2.1–2.2], one can directly obtain the results. $\square$

B.6.6 Proof of Thm. 125

We first establish the PV hyperplane equivalence and then derive the distance formula.

B.6.6.1 Equivalent Characterization of the PV Hyperplane

We first review a useful lemma from Chen et al. [55, Lem. J.1].

Lemma 196. *We assume that the manifold $\mathcal{M}$ admits a gyrogroup defined by*

$$x \oplus y = \text{Exp}_x(\text{PT}_{e \rightarrow x}(\text{Log}_e(y))), \forall x, y \in \mathcal{M}, \quad (\text{B.303})$$

where $e \in \mathcal{M}$ is the origin of the manifold. Then, we have the following

$$\langle \text{Log}_p(x), a \rangle_p = \langle \text{Log}_e(\ominus p \oplus x), \text{PT}_{p \rightarrow e}(a) \rangle_e, \quad \forall x, p \in \mathcal{M} \text{ and } \forall a \in T_p \mathcal{M}. \quad (\text{B.304})$$

Now, we are ready to prove Thm. 125.

Proof of PV hyperplane. Thm. 124 indicates that the assumption of Thm. 196 holds with $\mathcal{M} = \mathbb{P}\mathbb{V}_K^n$, $\oplus = \oplus_U$ and $e = \mathbf{0}$. Then, the PV hyperplane

$$H_{a,p} = \left\{ x \in \mathbb{P}\mathbb{V}_K^n \mid \langle \text{Log}_p(x), a \rangle_p = 0 \right\} \quad (\text{B.305})$$

can be rewritten as

$$H_{a,p} = \left\{ x \in \mathbb{P}\mathbb{V}_K^n \mid \langle \text{Log}_0(-p \oplus_U x), \text{PT}_{p \rightarrow 0}(a) \rangle_0 = 0 \right\}. \quad (\text{B.306})$$

Using the explicit PV operators in Thm. 123 and the PV metric in Eq. (5.1), we have

$$\begin{aligned} \text{Log}_0(-p \oplus_U x) &= \alpha(-p \oplus_U x), \quad \text{for some scalar } \alpha \geq 0, \\ \text{PT}_{p \rightarrow 0}(a) &= \beta d_p \pi(a), \quad \text{for some scalar } \beta > 0, \\ g_0(u, v) &= \langle u, v \rangle. \end{aligned} \quad (\text{B.307})$$

As $\alpha = 0$ is trivial, we only consider the case $\alpha > 0$:

$$\langle \text{Log}_0(-p \oplus_U x), \text{PT}_{p \rightarrow 0}(a) \rangle_0 = 0 \iff \langle -p \oplus_U x, d_p \pi(a) \rangle = 0. \quad (\text{B.308})$$

□

B.6.6.2 PV Point-to-Hyperplane Distance

We first prove a lemma on the isometry and point-to-hyperplane distance, which will be used to derive the PV point-to-hyperplane distance.

Lemma 197 (Isometry and point-to-hyperplane distance). *Let $(\mathcal{M}, g)$ and $(\bar{\mathcal{M}}, \bar{g})$ be Riemannian manifolds and let $\phi : \mathcal{M} \rightarrow \bar{\mathcal{M}}$ be a Riemannian isometry. For $p \in \mathcal{M}$ and $a \in T_p \mathcal{M}$, define the hyperplane*

$$H_{a,p} = \{x \in \mathcal{M} \mid g_p(\text{Log}_p(x), a) = 0\}. \quad (\text{B.309})$$

Let $\bar{p} = \phi(p)$ and $\bar{a} = d_p \phi(a) \in T_{\bar{p}} \bar{\mathcal{M}}$, and define the corresponding hyperplane on $\bar{\mathcal{M}}$ by

$$\bar{H}_{\bar{a}, \bar{p}} = \{\bar{x} \in \bar{\mathcal{M}} \mid \bar{g}_{\bar{p}}(\bar{\text{Log}}_{\bar{p}}(\bar{x}), \bar{a}) = 0\}. \quad (\text{B.310})$$

Then ϕ maps $H_{a,p}$ onto $\bar{H}_{\bar{a}, \bar{p}}$, that is,

$$\phi(H_{a,p}) = \bar{H}_{\bar{a}, \bar{p}}. \quad (\text{B.311})$$

Moreover, for every $x \in \mathcal{M}$ we have

$$d_{\mathcal{M}}(x, H_{a,p}) = d_{\bar{\mathcal{M}}}(\phi(x), \bar{H}_{\bar{a}, \bar{p}}), \quad (\text{B.312})$$

when the point-to-hyperplane distance exists. Here, $d_{\mathcal{M}}$ and $d_{\bar{\mathcal{M}}}$ denote the Riemannian distances on $\mathcal{M}$ and $\bar{\mathcal{M}}$, respectively.

Proof. Since ϕ is a Riemannian isometry, we have

$$g_p(\text{Log}_p(x), a) = \bar{g}_{\bar{p}}(d_p \phi(\text{Log}_p(x)), d_p \phi(a)) = \bar{g}_{\bar{p}}(\bar{\text{Log}}_{\bar{p}}(\phi(x)), \bar{a}). \quad (\text{B.313})$$

Therefore,

$$g_p(\text{Log}_p(x), a) = 0 \iff \bar{g}_{\bar{p}}(\bar{\text{Log}}_{\bar{p}}(\phi(x)), \bar{a}) = 0, \quad (\text{B.314})$$

which shows that $x \in H_{a,p}$ if and only if $\phi(x) \in \bar{H}_{\bar{a}, \bar{p}}$, and hence $\phi(H_{a,p}) = \bar{H}_{\bar{a}, \bar{p}}$.

For the point-to-hyperplane distance, recall that for a subset $S \subset \mathcal{M}$ the distance from x to S is

$$d_{\mathcal{M}}(x, S) = \inf_{z \in S} d_{\mathcal{M}}(x, z). \quad (\text{B.315})$$

For the point-to-hyperplane distance, we have

$$\begin{aligned}
 d_{\mathcal{M}}(x, H_{a,p}) &= \inf_{z \in H_{a,p}} d_{\mathcal{M}}(x, z) \\
 &= \inf_{z \in H_{a,p}} d_{\tilde{\mathcal{M}}}(\phi(x), \phi(z)) \\
 &= \inf_{\bar{z} \in \bar{H}_{\bar{a}, \bar{p}}} d_{\tilde{\mathcal{M}}}(\phi(x), \bar{z}) \\
 &= d_{\tilde{\mathcal{M}}}(\phi(x), \bar{H}_{\bar{a}, \bar{p}}).
 \end{aligned} \tag{B.316}$$

□

Next, we review the Poincaré hyperplane and point-to-hyperplane distance [76, Sec. 3.1].

Poincaré Point-to-Hyperplane Distance. For a point $p \in \mathbb{P}_K^n$ and a normal vector $a \in T_p \mathbb{P}_K^n$, the Poincaré point-to-hyperplane distance is given by Ganea et al. [76, Thm. 5]:

$$H_{a,p}^{\mathbb{P}} = \left\{ x \in \mathbb{P}_K^n \mid \langle \text{Log}_p^{\mathbb{P}}(x), a \rangle_p = 0 \right\} = \{ x \in \mathbb{P}_K^n \mid \langle -p \oplus_{\mathbb{M}} x, a \rangle = 0 \}, \tag{B.317}$$

$$d^{\mathbb{P}}(y, H_{a,p}^{\mathbb{P}}) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{2\sqrt{-K} |\langle -p \oplus_{\mathbb{M}} y, a \rangle|}{(1 + K \|-p \oplus_{\mathbb{M}} y\|^2) \|a\|} \right). \tag{B.318}$$

Proof of the PV point-to-hyperplane distance. Let

$$\bar{p} = \pi(p) \in \mathbb{P}_K^n, \quad \bar{a} = d_p \pi(a) \in T_{\bar{p}} \mathbb{P}_K^n, \quad \bar{y} = \pi(y) \in \mathbb{P}_K^n. \tag{B.319}$$

By Thm. 197, the point-to-hyperplane distances satisfy

$$d^{\mathbb{P}^V}(y, H_{a,p}) = d^{\mathbb{P}}(\bar{y}, \bar{H}_{\bar{a}, \bar{p}}). \tag{B.320}$$

Applying the Poincaré distance formula in Eq. (B.318) with $p = \bar{p}$, $a = \bar{a}$, and $y = \bar{y}$ gives

$$d^{\mathbb{P}}(\bar{y}, \bar{H}_{\bar{a}, \bar{p}}) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{2\sqrt{-K} |\langle -\bar{p} \oplus_{\mathbb{M}} \bar{y}, \bar{a} \rangle|}{(1 + K \|\bar{p} \oplus_{\mathbb{M}} \bar{y}\|^2) \|\bar{a}\|} \right). \tag{B.321}$$

The gyrovector isomorphism π implies

$$-\bar{p} \oplus_{\mathbb{M}} \bar{y} = \pi(-p \oplus_{\mathbb{U}} y). \tag{B.322}$$

Denote $z = -p \oplus_U y$. From Eq. (5.8), we have the explicit expression

$$\pi(z) = \frac{\beta_z}{1 + \beta_z} z, \quad (\text{B.323})$$

with $\beta_z > 0$. Since $\beta_z = \frac{1}{\sqrt{1-K\|z\|^2}}$, we obtain

$$\begin{aligned} 1 + K \|\pi(z)\|^2 &= 1 + K \left(\frac{\beta_z}{1 + \beta_z} \right)^2 \|z\|^2 \\ &= 1 + \frac{K \beta_z^2 \|z\|^2}{(1 + \beta_z)^2} \\ &= 1 + \frac{\beta_z^2 (1 - \beta_z^{-2})}{(1 + \beta_z)^2} \\ &= 1 + \frac{\beta_z^2 - 1}{(1 + \beta_z)^2} \\ &= 1 + \frac{\beta_z - 1}{1 + \beta_z} \\ &= \frac{2\beta_z}{1 + \beta_z} \end{aligned} \quad (\text{B.324})$$

The above yields

$$\frac{2\sqrt{-K} |\langle \pi(z), \bar{a} \rangle|}{(1 + K \|\pi(z)\|^2) \|\bar{a}\|} = \frac{\sqrt{-K} |\langle z, \bar{a} \rangle|}{\|\bar{a}\|}. \quad (\text{B.325})$$

Therefore,

$$d(y, H_{a,p}) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{\sqrt{-K} |\langle -p \oplus_U y, d_p \pi(a) \rangle|}{\|d_p \pi(a)\|} \right). \quad (\text{B.326})$$

□

B.6.7 Proof of Thm. 126

Proof of PV MLR. For clarity, we fix a class index k and omit k in the notation whenever possible. We denote $\pi = \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}$ as in Thm. 125.

Step 1: From Hyperplane Distance to a Signed Score. The PV MLR in

Eq. (5.24) associated with parameters (p, a) for $x \in \mathbb{P}\mathbb{V}_K^n$ is

$$\begin{aligned}
 v_k(x) &= \text{sign}(\langle -p_k \oplus_{\mathbb{U}} x, d_{p_k} \pi(a_k) \rangle) \|a_k\|_{p_k} d(x, H_{a_k, p_k}) \\
 &\stackrel{(1)}{=} \frac{\|a_k\|_{p_k}}{\sqrt{-K}} \text{sign}(\langle -p_k \oplus_{\mathbb{U}} x, d_{p_k} \pi(a_k) \rangle) \sinh^{-1} \left(\frac{\sqrt{-K} |\langle -p_k \oplus_{\mathbb{U}} x, d_{p_k} \pi(a_k) \rangle|}{\|d_{p_k} \pi(a_k)\|} \right) \\
 &\stackrel{(2)}{=} \frac{\|a_k\|_{p_k}}{\sqrt{-K}} \sinh^{-1} \left(\frac{\sqrt{-K} \langle -p_k \oplus_{\mathbb{U}} x, d_{p_k} \pi(a_k) \rangle}{\|d_{p_k} \pi(a_k)\|} \right).
 \end{aligned} \tag{B.327}$$

The above comes from the following.

- (1) Thm. 125;
- (2) $\sinh^{-1}$ is odd and strictly increasing.

Step 2: Trivialization and Reduction to a Single Direction. We adopt the unidirectional parameterization in Sec. 5.2.4.1:

$$p_k = \text{Exp}_{\mathbf{0}}(r_k[z_k]), \quad a_k = \text{PT}_{\mathbf{0} \rightarrow p_k}(z_k), \quad [z_k] = \frac{z_k}{\|z_k\|}, \tag{B.328}$$

with $z_k \in T_{\mathbf{0}}\mathbb{P}\mathbb{V}_K^n \cong \mathbb{R}^n$ and $r_k \in \mathbb{R}$.

As parallel transport is an isometry, we have

$$\|a_k\|_{p_k} = \|z_k\|_{\mathbf{0}} = \|z_k\|. \tag{B.329}$$

Moreover, p_k and z_k are collinear, because $\text{Exp}_{\mathbf{0}}$ in Thm. 123 preserves directions at the origin. Using the explicit expression of $\text{PT}_{\mathbf{0} \rightarrow y}$ at the origin in Thm. 123, we see that $\text{PT}_{\mathbf{0} \rightarrow p_k}$ maps z_k to a linear combination of z_k and p_k . Therefore, a_k is also collinear with z_k .

The differential $d_{p_k} \pi$ in Thm. 121 has the form

$$d_{p_k} \pi(v) = \alpha_k v + \beta_k \langle p_k, v \rangle p_k, \quad \alpha_k > 0, \beta_k \in \mathbb{R}, \tag{B.330}$$

so $d_{p_k} \pi$ maps any vector in $\text{span}\{z_k\}$ into the same one-dimensional subspace. Consequently, there exists a scalar $\lambda_k > 0$ such that

$$d_{p_k} \pi(a_k) = \lambda_k z_k. \tag{B.331}$$

The sign of λ_k can be absorbed into z_k by redefining $z_k \leftarrow -z_k$ if necessary. Without loss of generality we may assume $\lambda_k > 0$. Putting Eq. (B.328), Eq. (B.329), Eq. (B.331)

and $\|d_{p_k}\pi(a_k)\| = \lambda_k\|z_k\|$ into Eq. (B.327) yields

$$v_k(x) = \frac{\|z_k\|}{\sqrt{-K}} \sinh^{-1} \left(\frac{\sqrt{-K}}{\|z_k\|} \langle -p_k \oplus_U x, z_k \rangle \right). \quad (\text{B.332})$$

Step 3: Eliminating the Gyroaddition. The remaining task is to expand the gyro-additive term in Eq. (B.332). From Sec. 5.2.2, PV gyroaddition is given by

$$u \oplus_U v = u + v + \left(\frac{1 - \beta_v}{\beta_v} - K \frac{\beta_u}{1 + \beta_u} \langle u, v \rangle \right) u, \quad \beta_w = \frac{1}{\sqrt{1 - K\|w\|^2}}. \quad (\text{B.333})$$

Setting $u = -p_k$ and $v = x$ yields

$$-p_k \oplus_U x = -p_k + x + \left(\frac{1 - \beta_x}{\beta_x} - K \frac{\beta_{p_k}}{1 + \beta_{p_k}} \langle -p_k, x \rangle \right) (-p_k). \quad (\text{B.334})$$

Taking the inner product with z_k gives

$$\begin{aligned} \langle -p_k \oplus_U x, z_k \rangle &= \langle -p_k, z_k \rangle + \langle x, z_k \rangle + \left(\frac{1 - \beta_x}{\beta_x} - K \frac{\beta_{p_k}}{1 + \beta_{p_k}} \langle -p_k, x \rangle \right) \langle -p_k, z_k \rangle \\ &= \langle x, z_k \rangle + \left(1 + \frac{1 - \beta_x}{\beta_x} - K \frac{\beta_{p_k}}{1 + \beta_{p_k}} \langle -p_k, x \rangle \right) \langle -p_k, z_k \rangle. \end{aligned} \quad (\text{B.335})$$

Next, we rewrite the above expression using the unidirectional parameterization of p_k . From Eq. (B.328) and the explicit PV exponential at the origin in Thm. 123, we have

$$p_k = \text{Exp}_{\mathbf{0}}(r_k[z_k]) = \frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} r_k \right) \frac{z_k}{\|z_k\|}. \quad (\text{B.336})$$

Thus,

$$\langle -p_k, z_k \rangle = -\frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} r_k \right) \|z_k\|. \quad (\text{B.337})$$

Moreover, since p_k and z_k share the same direction, any x admits the decomposition

$$x = x_{\parallel} + x_{\perp}, \quad x_{\parallel} = \frac{\langle x, z_k \rangle}{\|z_k\|^2} z_k, \quad \langle x_{\perp}, z_k \rangle = 0, \quad (\text{B.338})$$

which implies

$$\langle -p_k, x \rangle = \langle -p_k, x_{\parallel} \rangle = \frac{\langle x, z_k \rangle}{\|z_k\|^2} \langle -p_k, z_k \rangle = -\frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} r_k \right) \frac{\langle x, z_k \rangle}{\|z_k\|}. \quad (\text{B.339})$$

The beta factor at p_k is

$$\beta_{p_k} = \frac{1}{\sqrt{1 - K\|p_k\|^2}} = \operatorname{sech}(\sqrt{-K}r_k), \quad (\text{B.340})$$

where we used $\|p_k\|^2 = -\frac{1}{K} \sinh^2(\sqrt{-K}r_k)$ and the identity $1 + \sinh^2(t) = \cosh^2(t)$.

Using $\langle -p_k, z_k \rangle$, $\langle -p_k, x \rangle$, and β_{p_k} , we have

$$\begin{aligned} & \langle -p_k \oplus_U x, z_k \rangle \\ &= \langle x, z_k \rangle + \left(1 + \frac{1 - \beta_x}{\beta_x} - K \frac{\beta_{p_k}}{1 + \beta_{p_k}} \langle -p_k, x \rangle \right) \langle -p_k, z_k \rangle \\ &= \langle x, z_k \rangle + \left(\frac{1}{\beta_x} - K \frac{\beta_{p_k}}{1 + \beta_{p_k}} \langle -p_k, x \rangle \right) \langle -p_k, z_k \rangle \\ &= \langle x, z_k \rangle + \frac{1}{\beta_x} \left(-\frac{\sinh(\sqrt{-K}r_k)}{\sqrt{-K}} \|z_k\| \right) \\ &\quad - K \frac{\beta_{p_k}}{1 + \beta_{p_k}} \left(-\frac{\sinh(\sqrt{-K}r_k)}{\sqrt{-K}} \frac{\langle x, z_k \rangle}{\|z_k\|} \right) \left(-\frac{\sinh(\sqrt{-K}r_k)}{\sqrt{-K}} \|z_k\| \right) \\ &= \langle x, z_k \rangle - \frac{\sinh(\sqrt{-K}r_k)}{\sqrt{-K}} \frac{\|z_k\|}{\beta_x} + \frac{\beta_{p_k} \sinh^2(\sqrt{-K}r_k)}{1 + \beta_{p_k}} \langle x, z_k \rangle \\ &= \left(1 + \frac{\beta_{p_k} \sinh^2(\sqrt{-K}r_k)}{1 + \beta_{p_k}} \right) \langle x, z_k \rangle - \frac{\sinh(\sqrt{-K}r_k)}{\sqrt{-K}} \frac{\|z_k\|}{\beta_x}. \end{aligned} \quad (\text{B.341})$$

Since $\beta_{p_k} = \operatorname{sech}(\sqrt{-K}r_k)$ and $1 + \sinh^2(\sqrt{-K}r_k) = \cosh^2(\sqrt{-K}r_k) = 1/\beta_{p_k}^2$, we have

$$\begin{aligned} 1 + \frac{\beta_{p_k} \sinh^2(\sqrt{-K}r_k)}{1 + \beta_{p_k}} &= \frac{1 + \beta_{p_k} + \beta_{p_k} \sinh^2(\sqrt{-K}r_k)}{1 + \beta_{p_k}} \\ &= \frac{1 + \beta_{p_k} \cosh^2(\sqrt{-K}r_k)}{1 + \beta_{p_k}} \\ &= \frac{1 + 1/\beta_{p_k}}{1 + \beta_{p_k}} = \frac{1}{\beta_{p_k}} = \cosh(\sqrt{-K}r_k), \end{aligned} \quad (\text{B.342})$$

which implies

$$\langle -p_k \oplus_U x, z_k \rangle = \cosh(\sqrt{-K}r_k) \langle x, z_k \rangle - \frac{\sinh(\sqrt{-K}r_k)}{\sqrt{-K}} \frac{\|z_k\|}{\beta_x}. \quad (\text{B.343})$$

Recalling that $\beta_x = 1/\sqrt{1-K\|x\|^2}$, we obtain

$$\langle -p_k \oplus_U x, z_k \rangle = \cosh(\sqrt{-K}r_k) \langle x, z_k \rangle - \frac{\sinh(\sqrt{-K}r_k)}{\sqrt{-K}} \|z_k\| \sqrt{1-K\|x\|^2}. \quad (\text{B.344})$$

Substituting Eq. (B.344) into Eq. (B.332), we arrive at

$$v_k(x) = \frac{\|z_k\|}{\sqrt{-K}} \sinh^{-1} \left(\cosh(\sqrt{-K}r_k) \frac{\sqrt{-K}}{\|z_k\|} \langle x, z_k \rangle - \sinh(\sqrt{-K}r_k) \sqrt{1-K\|x\|^2} \right). \quad (\text{B.345})$$

□

Proof of PV MLR limits. By Taylor expansions, we have

$$\begin{aligned} \cosh(\sqrt{-K}r_k) &= 1 - \frac{Kr_k^2}{2} + \mathcal{O}(K^2), \\ \sinh(\sqrt{-K}r_k) &= \sqrt{-K}r_k + \mathcal{O}((-K)^{3/2}), \\ \sqrt{1-K\|x\|^2} &= 1 - \frac{K\|x\|^2}{2} + \mathcal{O}(K^2). \end{aligned} \quad (\text{B.346})$$

The argument of $\sinh^{-1}(\cdot)$ in Eq. (B.345) can be simplified as

$$\begin{aligned} &\sinh^{-1} \left\{ \cosh(\sqrt{-K}r_k) \frac{\sqrt{-K}}{\|z_k\|} \langle x, z_k \rangle - \sinh(\sqrt{-K}r_k) \sqrt{1-K\|x\|^2} \right\} \\ &= \sinh^{-1} \left\{ \left(1 - \frac{Kr_k^2}{2} + \mathcal{O}(K^2) \right) \frac{\sqrt{-K}}{\|z_k\|} \langle x, z_k \rangle \right. \\ &\quad \left. - \left(\sqrt{-K}r_k + \mathcal{O}((-K)^{3/2}) \right) \left(1 - \frac{K\|x\|^2}{2} + \mathcal{O}(K^2) \right) \right\} \\ &= \sinh^{-1} \left\{ \sqrt{-K} \left(\frac{\langle x, z_k \rangle}{\|z_k\|} - r_k \right) + \mathcal{O}((-K)^{3/2}) \right\} \\ &= \sqrt{-K} \left(\frac{\langle x, z_k \rangle}{\|z_k\|} - r_k \right) + \mathcal{O}((-K)^{3/2}). \end{aligned} \quad (\text{B.347})$$

Substituting this into Eq. (B.345) gives

$$\begin{aligned} v_k(x) &= \frac{\|z_k\|}{\sqrt{-K}} \left(\sqrt{-K} \left(\frac{\langle x, z_k \rangle}{\|z_k\|} - r_k \right) + \mathcal{O}((-K)^{3/2}) \right) \\ &= \|z_k\| \left(\frac{\langle x, z_k \rangle}{\|z_k\|} - r_k \right) + \mathcal{O}(-K) \\ &= \langle x, z_k \rangle - r_k \|z_k\| + \mathcal{O}(-K), \\ &\xrightarrow{K \rightarrow 0^-} \langle x, z_k \rangle - r_k \|z_k\|. \end{aligned} \quad (\text{B.348})$$

□

B.6.8 Proof of Thm. 127

Proof of PV FC layer. Specializing Thm. 125 to $p = \mathbf{0}$ and $a = e_k$ and using that $-\mathbf{0} \oplus_{\mathbb{U}} y = y$ gives the LHS

$$\text{sign}(\langle d_{\mathbf{0}}\pi(e_k), -\mathbf{0} \oplus_{\mathbb{U}} y \rangle) d(y, H_{e_k, \mathbf{0}}) = \frac{1}{\sqrt{-K}} \sinh^{-1}(\sqrt{-K} y_k), \quad (\text{B.349})$$

with $y_k = \langle y, e_k \rangle$. Then, we obtain

$$y_k = \frac{1}{\sqrt{-K}} \sinh(\sqrt{-K} v_k(x)), \quad k = 1, \dots, m. \quad (\text{B.350})$$

□

Proof of PV FC limits. By Thm. 126, as $K \rightarrow 0^-$ we have

$$v_k(x) \rightarrow \langle x, z_k \rangle + b_k, \quad b_k = -r_k \|z_k\|. \quad (\text{B.351})$$

For $K < 0$ and $v_k(x) \neq 0$, we can rewrite y_k as

$$y_k = v_k(x) \frac{\sinh(\sqrt{-K} v_k(x))}{\sqrt{-K} v_k(x)}, \quad (\text{B.352})$$

and we define the fraction to be 1 when $v_k(x) = 0$. Since $\sqrt{-K} \rightarrow 0$ and $v_k(x)$ converges to a finite limit, we have $\sqrt{-K} v_k(x) \rightarrow 0$. Using the standard limit $\lim_{u \rightarrow 0} \sinh(u)/u = 1$, it follows that

$$\frac{\sinh(\sqrt{-K} v_k(x))}{\sqrt{-K} v_k(x)} \rightarrow 1 \quad \text{as} \quad K \rightarrow 0^-. \quad (\text{B.353})$$

Combining the above limits yields

$$\lim_{K \rightarrow 0^-} y_k = \lim_{K \rightarrow 0^-} v_k(x) = \langle x, z_k \rangle + b_k. \quad (\text{B.354})$$

□

B.6.9 Proof of Thm. 128

Proof. The result is first established in the Poincaré ball model in Thms. 83 and 90. Since $\pi = \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n} : \mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n$ is a Riemannian isometry, for $x, y \in \mathbb{P}\mathbb{V}_K^n$, $v \in T_x \mathbb{P}\mathbb{V}_K^n$,

and $t \in \mathbb{R}$, it intertwines the key geometric operators used in the proof:

$$\pi \left(\text{Exp}_x^{\mathbb{P}\mathbb{V}}(v) \right) = \text{Exp}_{\pi(x)}^{\mathbb{P}}(d_x \pi(v)), \quad (\text{B.355})$$

$$d_x \pi \left(\text{Log}_x^{\mathbb{P}\mathbb{V}}(y) \right) = \text{Log}_{\pi(x)}^{\mathbb{P}}(\pi(y)), \quad (\text{B.356})$$

$$\pi(x \oplus_{\mathbb{U}} y) = \pi(x) \oplus_{\mathbb{M}} \pi(y), \quad (\text{B.357})$$

$$\pi(t \otimes_{\mathbb{U}} x) = t \odot_{\mathbb{M}} \pi(x). \quad (\text{B.358})$$

Together with the preservation of geodesic distances and Fréchet means under the Riemannian isometry π , these identities imply that both homogeneity identities are preserved under π . Therefore the same theorem holds for the PV model by the isometry π . $\square$

B.6.10 Proof of Thm. 129

Proof. We first recall the isometries between the Poincaré ball and the hyperboloid [182, Sec. 2.1]:

$$\pi_{\mathbb{H}_K^n \rightarrow \mathbb{P}_K^n}(x) = \frac{x_s}{1 + \sqrt{|K|x_t}}, \quad (\text{B.359})$$

$$\pi_{\mathbb{P}_K^n \rightarrow \mathbb{H}_K^n}(y) = \begin{bmatrix} \frac{1}{\sqrt{|K|}} \frac{1 - K \|y\|^2}{1 + K \|y\|^2} \\ \frac{2y}{1 + K \|y\|^2} \end{bmatrix}. \quad (\text{B.360})$$

Hence, the following are Riemannian isometries:

$$\pi_{\mathbb{H}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n} = \pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n} \circ \pi_{\mathbb{H}_K^n \rightarrow \mathbb{P}_K^n}, \quad \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{H}_K^n} = \pi_{\mathbb{P}_K^n \rightarrow \mathbb{H}_K^n} \circ \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}_K^n}. \quad (\text{B.361})$$

It remains to derive the explicit formulas.

For $x = [x_t, x_s^\top]^\top \in \mathbb{H}_K^n$, we first map to the Poincaré ball:

$$y = \pi_{\mathbb{H}_K^n \rightarrow \mathbb{P}_K^n}(x) = \frac{x_s}{1 + \sqrt{|K|x_t}}. \quad (\text{B.362})$$

Applying $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}$ from Eq. (5.8) yields

$$\pi_{\mathbb{H}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(x) = \pi_{\mathbb{P}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(y) = 2\gamma_y^2 y, \quad \gamma_y = \frac{1}{\sqrt{1 + K \|y\|^2}}. \quad (\text{B.363})$$

Using $y = x_s/(1 + \sqrt{|K|x_t})$, we compute

$$1 + K \|y\|^2 = 1 + K \frac{\|x_s\|^2}{\left(1 + \sqrt{|K|x_t}\right)^2} = \frac{\left(1 + \sqrt{|K|x_t}\right)^2 + K \|x_s\|^2}{\left(1 + \sqrt{|K|x_t}\right)^2}. \quad (\text{B.364})$$

Since $x \in \mathbb{H}_K^n$ satisfies $\langle x, x \rangle_{\mathcal{L}} = 1/K$, we have

$$\langle x, x \rangle_{\mathcal{L}} = -x_t^2 + \|x_s\|^2 = \frac{1}{K} \quad \Rightarrow \quad \|x_s\|^2 = x_t^2 + \frac{1}{K}. \quad (\text{B.365})$$

Substituting this into the numerator gives

$$\begin{aligned} \left(1 + \sqrt{|K|x_t}\right)^2 + K \|x_s\|^2 &= \left(1 + \sqrt{|K|x_t}\right)^2 + K \left(x_t^2 + \frac{1}{K}\right) \\ &= \left(1 + \sqrt{|K|x_t}\right)^2 + K x_t^2 + 1 \\ &= 1 + 2\sqrt{|K|x_t} + |K|x_t^2 + K x_t^2 + 1 \\ &= 2 \left(1 + \sqrt{|K|x_t}\right). \end{aligned} \quad (\text{B.366})$$

Therefore,

$$1 + K \|y\|^2 = \frac{2 \left(1 + \sqrt{|K|x_t}\right)}{\left(1 + \sqrt{|K|x_t}\right)^2} = \frac{2}{1 + \sqrt{|K|x_t}}, \quad (\text{B.367})$$

and hence

$$\gamma_y^2 = \frac{1}{1 + K \|y\|^2} = \frac{1 + \sqrt{|K|x_t}}{2}. \quad (\text{B.368})$$

Finally,

$$\pi_{\mathbb{H}_K^n \rightarrow \mathbb{P}\mathbb{V}_K^n}(x) = 2\gamma_y^2 y = 2 \cdot \frac{1 + \sqrt{|K|x_t}}{2} \cdot \frac{x_s}{1 + \sqrt{|K|x_t}} = x_s. \quad (\text{B.369})$$

For $\pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{H}_K^n}$, take $x \in \mathbb{P}\mathbb{V}_K^n$ and map to the Poincaré ball by Eq. (5.8):

$$y = \pi_{\mathbb{P}\mathbb{V}_K^n \rightarrow \mathbb{P}^n_K}(x) = \frac{\beta_x}{1 + \beta_x} x, \quad \beta_x = \frac{1}{\sqrt{1 - K \|x\|^2}}. \quad (\text{B.370})$$

Applying $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{H}_K^n}$, we obtain

$$\pi_{\mathbb{P}_K^n \rightarrow \mathbb{H}_K^n}(x) = \pi_{\mathbb{P}_K^n \rightarrow \mathbb{H}_K^n}(y) = \begin{bmatrix} \frac{1}{\sqrt{|K|}} \frac{1 - K \|y\|^2}{1 + K \|y\|^2} \\ \frac{2y}{1 + K \|y\|^2} \end{bmatrix}. \quad (\text{B.371})$$

We now simplify the spatial and temporal components separately. We write

$$y = \frac{\beta_x}{1 + \beta_x} x, \quad \|y\|^2 = \left(\frac{\beta_x}{1 + \beta_x} \right)^2 \|x\|^2. \quad (\text{B.372})$$

Using $\beta_x^2 = 1/(1 - K \|x\|^2)$, we obtain

$$\begin{aligned} K \|y\|^2 &= K \|x\|^2 \frac{\beta_x^2}{(1 + \beta_x)^2} = \frac{\beta_x^2 - 1}{(1 + \beta_x)^2} = \frac{\beta_x - 1}{(1 + \beta_x)} \\ \Rightarrow 1 + K \|y\|^2 &= \frac{2\beta_x}{1 + \beta_x}, \quad 1 - K \|y\|^2 = \frac{2}{1 + \beta_x}. \end{aligned} \quad (\text{B.373})$$

The spatial component of $\pi_{\mathbb{P}_K^n \rightarrow \mathbb{H}_K^n}(x)$ is

$$\frac{2y}{1 + K \|y\|^2} = \frac{2 \frac{\beta_x}{1 + \beta_x} x}{\frac{2\beta_x}{1 + \beta_x}} = x, \quad (\text{B.374})$$

and the temporal component is

$$\frac{1}{\sqrt{|K|}} \frac{1 - K \|y\|^2}{1 + K \|y\|^2} = \frac{1}{\sqrt{|K|}} \cdot \frac{\frac{2}{1 + \beta_x}}{\frac{2\beta_x}{1 + \beta_x}} = \frac{1}{\sqrt{|K|}\beta_x} = \frac{\sqrt{1 - K \|x\|^2}}{\sqrt{|K|}} = \sqrt{\|x\|^2 - \frac{1}{K}}. \quad (\text{B.375})$$

Thus,

$$\pi_{\mathbb{P}_K^n \rightarrow \mathbb{H}_K^n}(x) = \begin{bmatrix} \sqrt{\|x\|^2 - \frac{1}{K}} \\ x \end{bmatrix}. \quad (\text{B.376})$$

□

B.7 Hyperbolic Busemann Neural Networks

B.7.1 Proof of Thm. 130

Proof. Denoting $\kappa^2 = -K$ with $\kappa > 0$, we rewrite the Busemann functions in Eqs. (5.40) and (5.41) as

$$\text{(Poincaré)} \quad B^v(x) = \frac{1}{\kappa} \log \left(\frac{\|v - \kappa x\|^2}{1 - \kappa^2 \|x\|^2} \right), \quad (\text{B.377})$$

$$\text{(Lorentz)} \quad B^v(x) = \frac{1}{\kappa} \log (\kappa x_t - \kappa \langle x_s, v \rangle). \quad (\text{B.378})$$

For the Poincaré case,

$$\frac{\|v - \kappa x\|^2}{1 - \kappa^2 \|x\|^2} = \frac{1 - 2\kappa \langle v, x \rangle + \kappa^2 \|x\|^2}{1 - \kappa^2 \|x\|^2} = 1 + \frac{-2\kappa \langle v, x \rangle + 2\kappa^2 \|x\|^2}{1 - \kappa^2 \|x\|^2}. \quad (\text{B.379})$$

Using $\log(1 + u) = u + O(u^2)$ as $u \rightarrow 0$ and $(1 - \kappa^2 \|x\|^2)^{-1} = 1 + O(\kappa^2)$, we obtain

$$\begin{aligned} B^v(x) &= \frac{1}{\kappa} [(-2\kappa \langle v, x \rangle + 2\kappa^2 \|x\|^2) (1 + O(\kappa^2)) + O(\kappa^2)] \\ &= -2 \langle v, x \rangle + 2\kappa \|x\|^2 + O(\kappa) \\ &= -2 \langle v, x \rangle + O(\kappa). \end{aligned} \quad (\text{B.380})$$

Therefore, $B^v(x) \xrightarrow{K \rightarrow 0^-} -2 \langle v, x \rangle$.

For the Lorentz case, the hyperboloid constraint $-x_t^2 + \|x_s\|^2 = -\kappa^{-2}$ and $x_t > 0$ yield

$$\kappa x_t = \sqrt{1 + \kappa^2 \|x_s\|^2} = 1 + \frac{1}{2} \kappa^2 \|x_s\|^2 + O(\kappa^4). \quad (\text{B.381})$$

Set $z = -\kappa \langle x_s, v \rangle + \frac{1}{2} \kappa^2 \|x_s\|^2 + O(\kappa^4)$. Then

$$\log(\kappa x_t - \kappa \langle x_s, v \rangle) = \log(1 + z) = z + O(z^2) = -\kappa \langle x_s, v \rangle + O(\kappa^2), \quad (\text{B.382})$$

which implies $B^v(x) = -\langle x_s, v \rangle + O(\kappa)$ and therefore $B^v(x) \xrightarrow{K \rightarrow 0^-} -\langle x_s, v \rangle$.

Substituting the above two limits into $u_k(x) = -\alpha_k B^{v_k}(x) + b_k$ gives the stated Euclidean limits of the logits. $\square$

B.7.2 Proof of Thm. 132

We first review two useful characterizations of Busemann functions in Hadamard spaces. The class $\mathcal{B}$ below is given by Bridson and Haefliger [29, p. 271]. The equivalence follows from Bridson and Haefliger [29, Prop. II.8.22], whose formulation in terms of Busemann functions uses the identification of horofunctions with Busemann functions up to additive constants in Bridson and Haefliger [29, Cor. II.8.20].

Definition 198. Let $(\mathcal{X}, d)$ be a Hadamard space, and let $\mathcal{B}$ be the set of functions $h : \mathcal{X} \rightarrow \mathbb{R}$ on $(\mathcal{X}, d)$ satisfying:

- (1) h is convex;
- (2) 1-Lipschitz: $|h(x) - h(y)| \leq d(x, y)$ for all $x, y \in \mathcal{X}$;
- (3) for any $x_0 \in \mathcal{X}$ and $r > 0$, the function h attains its minimum on the sphere $S_r(x_0)$ at a unique point y with $h(y) = h(x_0) - r$.

Proposition 199. For a function $h : \mathcal{X} \rightarrow \mathbb{R}$, the following conditions are equivalent:

- (1) h is a Busemann function;
- (2) $h \in \mathcal{B}$;
- (3) h is convex, and for every $t \in \mathbb{R}$, the set $h^{-1}(-\infty, t]$ is nonempty; moreover, for each $x \in \mathcal{X}$, the curve $c_x : [0, \infty) \rightarrow \mathcal{X}$ defined by $t \mapsto \pi_{h^{-1}(-\infty, h(x)-t]}(x)$ is a geodesic ray.

Now, we are ready to prove Thm. 132.

Proof. As $\tau_2 = \tau_1$ is trivial, we only consider $\tau_2 \neq \tau_1$. We assume $\tau_2 > \tau_1$, and discuss the other direction last.

Step 1: Symmetry. By definition,

$$d(H_{\tau_1}^\gamma, H_{\tau_2}^\gamma) = \inf_{x \in H_{\tau_1}^\gamma, y \in H_{\tau_2}^\gamma} d(x, y) = \inf_{y \in H_{\tau_2}^\gamma, x \in H_{\tau_1}^\gamma} d(y, x) = d(H_{\tau_2}^\gamma, H_{\tau_1}^\gamma). \quad (\text{B.383})$$

Step 2: Lower Bound. For any $x \in H_{\tau_2}^\gamma$ and $y \in H_{\tau_1}^\gamma$, the 1-Lipschitz property of B^γ gives

$$|B^\gamma(x) - B^\gamma(y)| \leq d(x, y). \quad (\text{B.384})$$

With $B^\gamma(x) = \tau_2$ and $B^\gamma(y) = \tau_1$ this yields

$$\tau_2 - \tau_1 \leq d(x, y) \quad \forall y \in H_{\tau_1}^\gamma. \quad (\text{B.385})$$

Taking infimum in Eq. (B.385) over $y \in H_{\tau_1}^\gamma$ gives

$$\tau_2 - \tau_1 \leq d(x, H_{\tau_1}^\gamma). \quad (\text{B.386})$$

Step 3: Upper Bound. For any $x \in H_{\tau_2}^\gamma$, by property (3) in Thm. 199, the projection map

$$c_x(t) = \pi_{\{B^\gamma \leq B^\gamma(x) - t\}}(x) = \pi_{HB_{\tau_2 - t}^\gamma}(x), \quad t \in [0, \infty), \quad (\text{B.387})$$

is a unit-speed geodesic ray: $d(x, c_x(t)) = t$.

Let $t = \tau_2 - \tau_1 > 0$ and $z = c_x(t) \in HB_{\tau_1}^\gamma$. If $B^\gamma(z) < \tau_1$, then z lies in the interior of the horoball $HB_{\tau_1}^\gamma$. We can move slightly from z toward x along the geodesic segment $\overline{xz}$ to obtain a point z_ε with $d(x, z_\varepsilon) < d(x, z)$, contradicting the minimality of the projection. Hence, the projected point $z = c_x(t)$ indeed lies on $H_{\tau_1}^\gamma$: $B^\gamma(z) = \tau_1$. Then, we have the following:

$$d(x, H_{\tau_1}^\gamma) \leq d(x, HB_{\tau_1}^\gamma) = d(x, z) = d(x, c_x(t)) = t = \tau_2 - \tau_1. \quad (\text{B.388})$$

Step 4: Sandwich Closure. Combining Eqs. (B.386) and (B.388) gives

$$d(x, H_{\tau_1}^\gamma) = \tau_2 - \tau_1. \quad (\text{B.389})$$

Combining Eq. (B.388) and Eq. (B.386),

$$d(x, H_{\tau_1}^\gamma) = \tau_2 - \tau_1, \quad \text{for every } x \in H_{\tau_2}^\gamma. \quad (\text{B.390})$$

The right-hand side does not depend on x , hence

$$d(H_{\tau_2}^\gamma, H_{\tau_1}^\gamma) = \tau_2 - \tau_1. \quad (\text{B.391})$$

Step 5: Opposite Direction. If $\tau_1 > \tau_2$, geodesic completeness of the Hadamard space allows the geodesic ray c_x from Step 3 to extend to a complete geodesic $\tilde{c}_x : \mathbb{R} \rightarrow \mathcal{X}$. For any $x \in H_{\tau_2}^\gamma$, let $s = \tau_1 - \tau_2$ and $z = \tilde{c}_x(-s)$. Since $B^\gamma(c_x(t)) = B^\gamma(x) - t$ for $t \geq 0$, convexity and the 1-Lipschitz property of B^γ give $B^\gamma(z) = B^\gamma(x) + s = \tau_1$.

Thus, $z \in H_{\tau_1}^\gamma$ and $d(x, z) = s$. Combining this upper bound with the 1-Lipschitz lower bound from Step 2 gives

$$d(x, H_{\tau_1}^\gamma) = \tau_1 - \tau_2, \quad \text{for every } x \in H_{\tau_2}^\gamma. \quad (\text{B.392})$$

Therefore, in both directions,

$$d(x, H_\tau^\gamma) = |B^\gamma(x) - \tau|, \quad \forall x \in \mathcal{X}, \quad (\text{B.393})$$

and the symmetry of the distance between horospheres gives

$$d(H_{\tau_2}^\gamma, H_{\tau_1}^\gamma) = |\tau_2 - \tau_1|. \quad (\text{B.394})$$

□

B.7.3 Proof of Thm. 135

Proof. The hyperplane and point-to-hyperplane distance are presented in Tabs. A.16 and A.17. The origin of the Poincaré ball model is $e = \mathbf{0} \in \mathbb{P}_K^m$. The specific ones w.r.t. the origin [181, Def. 1 and Eq. (56)] are

$$H_{e_k, \mathbf{0}} = \{y \in \mathbb{P}_K^m \mid \langle e_k, y \rangle = 0\}, \quad (\text{B.395})$$

$$\bar{d}(y, H_{e_k, \mathbf{0}}) = \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\frac{2\sqrt{-K}y_k}{1 + K\|y\|^2} \right). \quad (\text{B.396})$$

Equating $\bar{d}(y, H_{e_k, \mathbf{0}})$ with $u_k(x)$ from Eq. (5.59) gives

$$\sinh^{-1} \left(\frac{2\sqrt{-K}y_k}{1 + K\|y\|^2} \right) = \sqrt{-K}u_k(x), \quad \forall k \in \{1, \dots, m\}. \quad (\text{B.397})$$

Note that Eq. (B.397) takes the same form as Shimizu et al. [181, Eq. (56)], except that their responses are different. The proof below is inspired by their derivation.

Applying $\sinh(\cdot)$ on both sides of Eq. (B.397) yields

$$\frac{2\sqrt{-K}y_k}{1 + K\|y\|^2} = \sinh \left(\sqrt{-K}u_k(x) \right). \quad (\text{B.398})$$

Define $\omega_k := \frac{1}{\sqrt{-K}} \sinh(\sqrt{-K} u_k(x))$ and $\omega = [\omega_k]_{k=1}^m$. Then, we have

$$2y_k = (1 + K \|y\|^2) \omega_k, \quad \forall k, \quad (\text{B.399})$$

which is equivalent to the vector identity

$$2y = (1 + K \|y\|^2) \omega. \quad (\text{B.400})$$

Hence, y is collinear with ω . Write $y = \lambda \omega$ with $\lambda \geq 0$. Substituting into Eq. (B.400) and taking norms gives a quadratic in λ :

$$K \|\omega\|^2 \lambda^2 - 2\lambda + 1 = 0. \quad (\text{B.401})$$

Solving and selecting the branch that satisfies $y \rightarrow 0$ as $\omega \rightarrow 0$ yields

$$\lambda = \frac{1 - \sqrt{1 - K \|\omega\|^2}}{K \|\omega\|^2} = \frac{1}{1 + \sqrt{1 - K \|\omega\|^2}}. \quad (\text{B.402})$$

Therefore,

$$y = \frac{\omega}{1 + \sqrt{1 - K \|\omega\|^2}}, \quad \omega_k = \frac{\sinh(\sqrt{-K} u_k(x))}{\sqrt{-K}}, \quad (\text{B.403})$$

which proves the claim. One can check that $y \in \mathbb{P}_K^m$. $\square$

B.7.4 Proof of Thm. 136

Proof. Recalling Tab. A.16, a Lorentz hyperplane is

$$H_{w,p} = \{x \in \mathbb{L}_K^m \mid \langle w, x \rangle_{\mathcal{L}} = 0\}, \quad \text{with } p \in \mathbb{L}_K^m, \quad w \in T_p \mathbb{L}_K^m. \quad (\text{B.404})$$

The canonical origin is $\bar{\mathbf{0}} \in \mathbb{L}_K^m$. The tangent space at the origin is

$$T_{\bar{\mathbf{0}}} \mathbb{L}_K^m = \{[0, v^\top]^\top \mid v \in \mathbb{R}^m\}, \quad (\text{B.405})$$

where each tangent vector has a zero time component. Therefore, the coordinate hyperplane through the origin and orthogonal to the k -th axis is

$$\begin{aligned} H_{\bar{e}_k, e} &= \{y \in \mathbb{L}_K^m \mid \langle \bar{e}_k, y \rangle_{\mathcal{L}} = 0\} \\ &= \{y = (y_t, y_s) \in \mathbb{L}_K^m \mid (y_s)_k = 0\}, \end{aligned} \quad (\text{B.406})$$

where $\bar{e}_k = [0, e_k^\top]^\top \in T_{\bar{\mathbf{0}}} \mathbb{L}_K^m$. From Tab. A.17, the associated signed point-to-hyperplane distance is

$$\begin{aligned} \bar{d}(y, H_{\bar{e}_k, e}) &= \text{sign}(\langle \bar{e}_k, y \rangle_{\mathcal{L}}) d(y, H_{\bar{e}_k, e}) \\ &= \frac{1}{\sqrt{-K}} \sinh^{-1} \left(\sqrt{-K} (y_s)_k \right). \end{aligned} \quad (\text{B.407})$$

Equating $\bar{d}(y, H_{\bar{e}_k, e})$ with $u_k(x)$ from Eq. (5.59) gives

$$\sinh^{-1} \left(\sqrt{-K} (y_s)_k \right) = \sqrt{-K} u_k(x), \quad 1 \leq k \leq m. \quad (\text{B.408})$$

Applying $\sinh(\cdot)$ to both sides of Eq. (B.408) yields

$$(y_s)_k = \frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} u_k(x) \right), \quad 1 \leq k \leq m. \quad (\text{B.409})$$

Stacking the coordinates gives

$$y_s = \frac{1}{\sqrt{-K}} \sinh \left(\sqrt{-K} u(x) \right), \quad u(x) = (u_1(x), \dots, u_m(x))^\top. \quad (\text{B.410})$$

Since $y \in \mathbb{L}_K^m$, the hyperboloid constraint $\langle y, y \rangle_{\mathcal{L}} = 1/K$ implies $-y_t^2 + \|y_s\|^2 = 1/K$. Taking the positive time component yields

$$y_t = \sqrt{\frac{1}{-K} + \|y_s\|^2}. \quad (\text{B.411})$$

Combining the expressions for y_t and y_s proves the claim. $\square$

B.7.5 Proof of Thm. 137

Proof. Set $K = -\kappa^2$ with $\kappa > 0$.

Poincaré Case. Recall that

$$y = \frac{\omega}{1 + \sqrt{1 + \kappa^2 \|\omega\|^2}}, \quad \omega_k = \frac{\sinh(\kappa u_k(x))}{\kappa}. \quad (\text{B.412})$$

For any bounded scalar z ,

$$\sinh(\kappa z) = \kappa z + \frac{\kappa^3 z^3}{3!} + O(\kappa^5) \Rightarrow \frac{\sinh(\kappa z)}{\kappa} = z + \frac{\kappa^2 z^3}{6} + O(\kappa^4). \quad (\text{B.413})$$

Applying this to $z = u_k(x)$ yields

$$\omega_k = u_k(x) + \frac{\kappa^2}{6} u_k(x)^3 + O(\kappa^4) = u_k(x) + O(\kappa^2). \quad (\text{B.414})$$

Thus, $\omega = u(x) + O(\kappa^2)$ and $\|\omega\|^2 = \|u(x)\|^2 + O(\kappa^2)$.

For the denominator,

$$\sqrt{1 + \kappa^2 \|\omega\|^2} = 1 + \frac{1}{2} \kappa^2 \|\omega\|^2 + O(\kappa^4), \quad (\text{B.415})$$

which gives

$$1 + \sqrt{1 + \kappa^2 \|\omega\|^2} = 2 + \frac{1}{2} \kappa^2 \|\omega\|^2 + O(\kappa^4). \quad (\text{B.416})$$

Taking the reciprocal produces

$$\frac{1}{1 + \sqrt{1 + \kappa^2 \|\omega\|^2}} = \frac{1}{2} + O(\kappa^2). \quad (\text{B.417})$$

Multiplying with $\omega = u(x) + O(\kappa^2)$ gives

$$y = \frac{1}{2} u(x) + O(\kappa^2). \quad (\text{B.418})$$

By Thm. 130, $u_k(x) \rightarrow 2\alpha_k \langle v_k, x \rangle + b_k$, hence

$$y_k \rightarrow \alpha_k \langle v_k, x \rangle + \frac{1}{2} b_k. \quad (\text{B.419})$$

Lorentz Case. Recall that

$$y_s = \frac{1}{\kappa} \sinh(\kappa u(x)), \quad y_t = \sqrt{\frac{1}{\kappa^2} + \|y_s\|^2}. \quad (\text{B.420})$$

Using the same expansion as above,

$$y_{s,k} = \frac{1}{\kappa} \left(\kappa u_k(x) + \frac{\kappa^3}{3!} u_k(x)^3 + O(\kappa^5) \right) = u_k(x) + O(\kappa^2). \quad (\text{B.421})$$

Therefore, $y_s = u(x) + O(\kappa^2)$ and $\|y_s\|^2 = \|u(x)\|^2 + O(\kappa^2)$.

Factor out κ^{-1} and expand the square root:

$$\begin{aligned}
 y_t &= \frac{1}{\kappa} \sqrt{1 + \kappa^2 \|y_s\|^2} \\
 &= \frac{1}{\kappa} \left(1 + \frac{1}{2} \kappa^2 \|y_s\|^2 + O(\kappa^4) \right) \\
 &= \frac{1}{\kappa} + \frac{\kappa}{2} \|y_s\|^2 + O(\kappa^3) \\
 &= \frac{1}{\kappa} + O(\kappa) \rightarrow \infty.
 \end{aligned} \tag{B.422}$$

By Thm. 130, $u_k(x) \rightarrow \alpha_k \langle v_k, x_s \rangle + b_k$. Using the spatial expansion yields

$$(y_s)_k \rightarrow \alpha_k \langle v_k, x_s \rangle + b_k. \tag{B.423}$$

This completes the proof. □

B.8 Full-Rank Correlation Networks

B.8.1 Proof of Thm. 138

We first prove a lemma for MLRs on general isometric manifolds, of which this theorem is a specific case. Notably, the result and proof can be readily extended to the case where $\mathbb{R}^m$ is endowed with an arbitrary inner product.

Lemma 200 (Isometric Riemannian MLRs). *Given m -dimensional Riemannian manifolds $(\widetilde{\mathcal{M}}, g^{\widetilde{\mathcal{M}}})$ and $(\mathcal{M}, g^{\mathcal{M}})$ with a Riemannian isometry $\phi : \widetilde{\mathcal{M}} \rightarrow \mathcal{M}$, their origins are $E \in \widetilde{\mathcal{M}}$ and $\phi(E) \in \mathcal{M}$. The Riemannian MLR over $\widetilde{\mathcal{M}}$ for the input $X \in \widetilde{\mathcal{M}}$ of each class $k = 1, \dots, C$ can be calculated by the one over $\mathcal{M}$:*

$$v_k^{\widetilde{\mathcal{M}}}(X; Z_k, \gamma_k) = v_k^{\mathcal{M}}(\phi(X); \phi_{*,E}(Z_k), \gamma_k), \tag{B.424}$$

with $\gamma_k \in \mathbb{R}$, $Z_k \in T_E \widetilde{\mathcal{M}} \cong \mathbb{R}^m$, and $\phi_{,E} : T_E \widetilde{\mathcal{M}} \rightarrow T_{\phi(E)} \mathcal{M}$ as the differential map. Here, $v_k^{\widetilde{\mathcal{M}}}$ and $v_k^{\mathcal{M}}$ are the specific realizations of the Riemannian MLR reviewed in Sec. 4.3.2.1 over $\widetilde{\mathcal{M}}$ and $\mathcal{M}$, respectively.*

Proof. We omit the subscript k in A_k and P_k for simplicity. We denote $\widetilde{\Gamma}$, $\widetilde{\text{Log}}$, $\langle \cdot, \cdot \rangle_P$, $\|\cdot\|_P$, $\widetilde{\text{d}}(X, \widetilde{H}_{A,P})$, $\widetilde{H}_{A,P}$ as the parallel transport along the geodesic, Riemannian logarithm, Riemannian metric, the induced norm, margin distance and hyperplane over

$\widetilde{\mathcal{M}}$, while the counterparts over $\mathcal{M}$ are denoted as Γ , Log , $\langle \cdot, \cdot \rangle_{\phi(P)}$, $\|\cdot\|_{\phi(P)}$, d , and H , respectively.

From the isometry, we have

$$\|A\|_P = \|\phi_{*,P}(A)\|_{\phi(P)}, \quad (\text{B.425})$$

$$\left\langle \widetilde{\text{Log}}_P(X), A \right\rangle_P = \left\langle \text{Log}_{\phi(P)}(\phi(X)), \phi_{*,P}(A) \right\rangle_{\phi(P)}. \quad (\text{B.426})$$

The above equations imply

$$\phi\left(\widetilde{H}_{A,P}\right) = H_{\phi_{*,P}(A),\phi(P)}. \quad (\text{B.427})$$

Denoting $\mathcal{H} = H_{\phi_{*,P}(A),\phi(P)}$, we have the following for the margin distance

$$\begin{aligned} \widetilde{d}(X, \widetilde{H}_{A,P}) &= \inf_{Q \in \widetilde{H}_{A,P}} \widetilde{d}(X, Q) \\ &\stackrel{(1)}{=} \inf_{Q \in \widetilde{H}_{A,P}} d(\phi(X), \phi(Q)) \\ &\stackrel{(2)}{=} \inf_{R \in \mathcal{H}} d(\phi(X), R) \\ &\stackrel{(3)}{=} d(\phi(X), \mathcal{H}). \end{aligned} \quad (\text{B.428})$$

The above comes from the following.

- (1) Isometry.
- (2) Eq. (B.427).
- (3) Definition of margin distance.

Combining the above, we have

$$\begin{aligned} v^{\widetilde{\mathcal{M}}}(X; P, A) &= \text{sign}(\langle A, \widetilde{\text{Log}}_P(X) \rangle_P) \|A\|_P \widetilde{d}(X, \widetilde{H}_{A,P}) \\ &= \text{sign}\left(\left\langle \text{Log}_{\phi(P)}(\phi(X)), \phi_{*,P}(A) \right\rangle_{\phi(P)}\right) \|\phi_{*,P}(A)\|_{\phi(P)} d(\phi(X), H_{\phi_{*,P}(A),\phi(P)}) \\ &= v^{\mathcal{M}}(\phi(X); \phi(P), \phi_{*,P}(A)). \end{aligned} \quad (\text{B.429})$$

Finally, let us further consider trivialization. By isometry, we have the following:

$$\begin{aligned} A &= \widetilde{\Gamma}_{E \rightarrow P}(Z) \\ &= \phi_{*,P}^{-1}\left(\text{PT}_{\phi(E) \rightarrow \phi(P)}(\phi_{*,E}(Z))\right), \end{aligned} \quad (\text{B.430})$$

$$\begin{aligned}
 P &= \widetilde{\text{Exp}}_E(\gamma[Z]) \\
 &= \phi^{-1}(\text{Exp}_{\phi(E)}(\gamma[\phi_{*,E}(Z)])) .
 \end{aligned} \tag{B.431}$$

Then, we have

$$\phi_{*,P}(A) = \text{PT}_{\phi(E) \rightarrow \phi(P)}(\phi_{*,E}(Z)), \tag{B.432}$$

$$\phi(P) = \text{Exp}_{\phi(E)}(\gamma[\phi_{*,E}(Z)]). \tag{B.433}$$

Putting the above two equations into Eq. (B.429), we have

$$\begin{aligned}
 v^{\widetilde{\mathcal{M}}}(X; Z, \gamma) &= v^{\widetilde{\mathcal{M}}}(X; P, A) \\
 &= v^{\mathcal{M}}(\phi(X); \phi(P), \phi_{*,P}(A)) \\
 &= v^{\mathcal{M}}(\phi(X); \text{Exp}_{\phi(E)}(\gamma[\phi_{*,E}(Z)]), \text{PT}_{\phi(E) \rightarrow \phi(P)}(\phi_{*,E}(Z))) \\
 &= v_k^{\mathcal{M}}(\phi(X); \phi_{*,E}(Z_k), \gamma_k).
 \end{aligned} \tag{B.434}$$

□

Thm. 138 is a special case of Thm. 200 and can be readily proven accordingly.

Proof of Thm. 138. MLR. In Euclidean space $\mathbb{R}^m$, simple computations show that the Riemannian MLR reviewed in Sec. 4.3.2.1 becomes Eq. (4.1), where the latter is equal to $\langle a_k, x - p_k \rangle$. Based on Thm. 200, we have

$$\begin{aligned}
 v_k(X; Z_k, \gamma_k) &= v_k^{\mathbb{R}^m}(\phi(X); \phi_{*,E}(Z_k), \gamma_k), \\
 &= \langle \phi(X) - \gamma_k[\phi_{*,E}(Z_k)], \phi_{*,E}(Z_k) \rangle \\
 &= \langle \phi(X), \phi_{*,E}(Z_k) \rangle - \gamma_k \|\phi_{*,E}(Z_k)\| ,
 \end{aligned} \tag{B.435}$$

Margin Hyperplane. In Euclidean space $\mathbb{R}^m$, the Riemannian margin hyperplane becomes the Euclidean one, which is parameterized by $\langle a_k, x - p_k \rangle = 0$. Together with Eq. (B.435), the results can be easily obtained. □

B.8.2 Proof of Thm. 139

Proof. First, we have the following:

$$\Theta(I) = I, \tag{B.436}$$

$$\text{Chol}(I) = I, \tag{B.437}$$

$$\log_{*,I}(V) = V, \quad \forall V \in \text{Hol}(n), \tag{B.438}$$

$$\log_{*,I}(V) = V, \quad \forall V \in \text{LT}^0(n), \quad (\text{B.439})$$

$$\mathcal{D}^*(I) = I. \quad (\text{B.440})$$

Putting the above into the differential formulas collected in Sec. 2.9.2, one can directly get the result w.r.t. ECM, LECM, and OLM. For LSM, based on Sec. 2.9.2, we have

$$\begin{aligned} \text{Log}_{*,I}^*(V) &= \log_{*,\Sigma} \left(\Delta V \Delta + \frac{1}{2} (V^0 \Sigma + \Sigma V^0) \right) \\ &\stackrel{(1)}{=} V + \frac{1}{2} (V^0 + V^0) \\ &\stackrel{(2)}{=} V - \text{diag}(V \mathbf{1}). \end{aligned} \quad (\text{B.441})$$

The above comes from the following.

$$(1) \quad \Sigma = \Delta = I$$

$$(2)$$

$$\begin{aligned} V^0 &= -2 \text{diag}((I_n + \Sigma)^{-1} \Delta V \Delta \mathbf{1}) \\ &= -\text{diag}(V \mathbf{1}) \end{aligned} \quad (\text{B.442})$$

□

B.8.3 Proof of Thm. 142

Let $d_n = \frac{n(n-1)}{2}$ and $d_m = \frac{m(m-1)}{2}$ be the manifold dimensions of $\text{Cor}^+(n)$ and $\text{Cor}^+(m)$, respectively. We have the following general results.

Lemma 201. *Let $(\text{Cor}^+(n), g^n)$ be isometric to $\mathbb{R}^{d_n}$ by the diffeomorphism*

$$\phi_n : \text{Cor}^+(n) \rightarrow \mathbb{R}^{d_n}, \quad (\text{B.443})$$

and let $(\text{Cor}^+(m), g^m)$ be isometric to $\mathbb{R}^{d_m}$ by the diffeomorphism

$$\phi_m : \text{Cor}^+(m) \rightarrow \mathbb{R}^{d_m}. \quad (\text{B.444})$$

The diffeomorphism satisfies

$$I_n = \phi_n^{-1}(\mathbf{0}_{d_n}), \quad I_m = \phi_m^{-1}(\mathbf{0}_{d_m}). \quad (\text{B.445})$$

The correlation FC layer $\mathcal{F} : \text{Cor}^+(n) \rightarrow \text{Cor}^+(m)$ for the input $X \in \text{Cor}^+(n)$ is

$$Y = \phi_m^{-1} \left(\sum_{i=1}^{d_m} v_i(X) e_i \right), \quad (\text{B.446})$$

where $\{e_i\}_{i=1}^{d_m}$ is the canonical orthonormal basis over $\mathbb{R}^{d_m}$ with $e_i = (\delta_{ik})_{k=1}^{d_m}$ for each i . Here, $\{v_i(X)\}_{i=1}^{d_m}$ is given by Thm. 138:

$$v_i(X) = \langle \phi_n(X), (\phi_n)_{*,I_n}(Z_i) \rangle - \gamma_i \|(\phi_n)_{*,I_n}(Z_i)\|, \quad (\text{B.447})$$

with $Z_i \in T_{I_n} \text{Cor}^+(n) \cong \mathbb{R}^{d_n}$ and $\gamma_i \in \mathbb{R}$ as the FC parameters.

Proof of Thm. 201. Let $\{O_k = (\phi_m)_{*,I_m}^{-1}(e_k)\}_{k=1}^{d_m}$. Then $\{O_k\}_{k=1}^{d_m}$ is an orthonormal basis over $T_{I_m} \text{Cor}^+(m)$.

The LHS of Eq. (5.69) is

$$\begin{aligned} & \text{sign} \left(\langle \text{Log}_{I_m}(Y), O_k \rangle_{I_m} \right) d(Y, H_{O_k, I_m}) \\ & \stackrel{(1)}{=} \text{sign} \left(\langle (\phi_m)_{*,I_m}^{-1} \phi_m(Y), O_k \rangle_{I_m} \right) d(Y, H_{O_k, I_m}) \\ & \stackrel{(2)}{=} \text{sign} (\langle \phi_m(Y), e_k \rangle) d(Y, H_{O_k, I_m}) \\ & \stackrel{(3)}{=} \text{sign} (\langle \phi_m(Y), e_k \rangle) d(\phi_m(Y), H_{e_k, \mathbf{0}_{d_m}}) \\ & = (\phi_m(Y))_k, \end{aligned} \quad (\text{B.448})$$

where (1)–(2) come from the isometry, and (3) comes from Eq. (B.428).

The RHS of Eq. (5.69) can be implied by Thm. 138. $\square$

Thm. 201 can be naturally extended to the cases where the inner products of $\mathbb{R}^{d_n}$ and $\mathbb{R}^{d_m}$ are not canonical.

Lemma 202. *Following all the notation in Thm. 201, we further assume that the inner products $Q^n(\cdot, \cdot)$ over $\mathbb{R}^{d_n}$ and $Q^m(\cdot, \cdot)$ over $\mathbb{R}^{d_m}$ are not necessarily canonical. In addition, $f : (\mathbb{R}^{d_m}, Q^m(\cdot, \cdot)) \rightarrow (\mathbb{R}^{d_m}, \langle \cdot, \cdot \rangle)$ is a linear isometry to the canonical inner product. Then, we have*

$$Y = \phi_m^{-1} \circ f^{-1} \left(\sum_{i=1}^{d_m} v_i(X) e_i \right), \quad (\text{B.449})$$

$$v_i(X) = Q^n(\phi_n(X), (\phi_n)_{*,I_n}(Z_i)) - \gamma_i \|(\phi_n)_{*,I_n}(Z_i)\|^{Q^n}, \quad (\text{B.450})$$

$$\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ l_{21} & 0 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{m1} & l_{m2} & l_{m3} & \cdots & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & \star & \star & \cdots & \star \\ h_{21} & 0 & \star & \cdots & \star \\ h_{31} & h_{32} & 0 & \cdots & \star \\ \vdots & \vdots & \vdots & \ddots & \star \\ h_{m1} & h_{m2} & h_{m3} & \cdots & 0 \end{pmatrix} \quad \begin{pmatrix} r_{11} & \star & \cdots & \star \\ r_{21} & r_{22} & \cdots & \star \\ \vdots & \vdots & \ddots & \star \\ r_{m-1,1} & r_{m-1,2} & \cdots & \star \\ -\sum_{i=1}^{m-1} r_{i1} & -\sum_{i=1}^{m-1} r_{i2} & \cdots & \sum_{j=1}^{m-1} \sum_{i=1}^{m-1} r_{ij} \end{pmatrix}$$

$L \in \text{LT}^0(m) \qquad H \in \text{Hol}(m) \qquad R \in \text{Row}_0(m)$

Figure B.1: Illustration of the Euclidean spaces $\text{LT}^0(m)$, $\text{Hol}(m)$ and $\text{Row}_0(m)$, where $\star$ can be obtained by symmetry.

where $\|\cdot\|^{Q^n}$ is the norm induced by Q^n .

Proof of Thm. 202. First, we denote

$$\psi^m = f \circ \phi_m : (\text{Cor}^+(m), g^m) \rightarrow (\mathbb{R}^{d_m}, \langle \cdot, \cdot \rangle). \quad (\text{B.451})$$

Note that the differential of any linear map between vector spaces is itself. The rest of the proof is identical to that of Thm. 201. $\square$

Now, we present the proof of Thm. 142.

Proof of Thm. 142. As ECM, LECM, OLM, and LSM are pullback metrics from Euclidean spaces, we resort to Thm. 201 and its extension Thm. 202. Denoting the zero matrix as $\mathbf{0}$, we have the following:

$$\phi^{\text{EC}}(I_n) = \log \circ \Theta(I_n) = \mathbf{0} \in \text{LT}^0(n), \quad (\text{B.452})$$

$$\text{Log}^\circ(I_n) = \mathbf{0} \in \text{Hol}(n), \quad (\text{B.453})$$

$$\text{Log}^\star(I_n) = \mathbf{0} \in \text{Row}_0(n). \quad (\text{B.454})$$

Therefore, the identity matrix is indeed the origin defined in Thm. 201.

Recalling Thm. 201, the prototype space is the vector space with the standard vector inner product. Obviously, $\text{LT}^0(m)$, $\text{Hol}(m)$, and $\text{Row}_0(m)$ are linearly isomorphic to $\mathbb{R}^{m(m-1)/2}$. As shown in Fig. B.1, each $L \in \text{LT}^0(m)$ can be identified with a vector of its lower triangular part. Besides, $\text{LT}^0(m)$ with the canonical matrix inner product is identified with $\mathbb{R}^{\frac{m(m-1)}{2}}$ with standard vector inner product. Therefore, the basis over $\text{LT}^0(m)$ corresponding to the canonical orthonormal basis over $\mathbb{R}^{m(m-1)/2}$ is

$$(\text{LT}^0(m), \langle \cdot, \cdot \rangle) : U_{ij}^{\text{LT}^0(m)} = E_{ij}, \quad 1 \leq j < i \leq m, \quad (\text{B.455})$$

where $E_{ij} \in \mathbb{R}^{m \times m}$ is the standard basis matrix, with the (k, l) -th element defined as

$$(E_{ij})_{kl} = \begin{cases} 1 & \text{if } k = i \text{ and } l = j, \\ 0 & \text{otherwise.} \end{cases} \quad (\text{B.456})$$

Without loss of generality, we identify $(\text{LT}^0(m), \langle \cdot, \cdot \rangle)$ with $(\mathbb{R}^{\frac{m(m-1)}{2}}, \langle \cdot, \cdot \rangle)$, and refer to $\{E_{ij}\}_{1 \leq j < i \leq m}$ as the canonical orthonormal basis.

However, $\{E_{ij}\}$ is neither a canonical orthonormal basis nor even orthonormal for $\text{Hol}(m)$ and $\text{Row}_0(m)$ under the standard matrix inner product. According to Thm. 202, we only need to find the linear isometry that maps these two spaces into $(\text{LT}^0(m), \langle \cdot, \cdot \rangle)$. By Fig. B.1, we have the following linear isometries to pull back these two inner products to the standard ones over $\text{LT}^0(m)$:

$$\begin{aligned} f_{\text{Hol}(m) \rightarrow \text{LT}^0(m)} : (\text{Hol}(m), \langle \cdot, \cdot \rangle) &\rightarrow (\text{LT}^0(m), \langle \cdot, \cdot \rangle), \\ \text{Hol}(m) \ni H &\mapsto \sqrt{2}[H] \in \text{LT}^0(m), \\ f_{\text{Row}_0(m) \rightarrow \text{LT}^0(m)} : (\text{Row}_0(m), \langle \cdot, \cdot \rangle) &\rightarrow (\text{LT}^0(m), \langle \cdot, \cdot \rangle), \\ \text{Row}_0(m) \ni R &\mapsto \sqrt{6}[\tilde{R}] + \sqrt{3}\mathbb{D}(\tilde{R}) \in \text{LT}^0(m), \end{aligned} \quad (\text{B.457})$$

where $\tilde{R} \in \mathcal{S}^{m-1}$ is the leading principal submatrix of order $m-1$ of R . The bases $f_{\text{Hol}(m) \rightarrow \text{LT}^0(m)}^{-1}(\{E_{ij}\})$ and $f_{\text{Row}_0(m) \rightarrow \text{LT}^0(m)}^{-1}(\{E_{ij}\})$ are as follows:

$$(\text{Hol}(m), \langle \cdot, \cdot \rangle) : U_{ij}^{\text{Hol}(m)} = \frac{E_{ij} + E_{ji}}{\sqrt{2}}, \quad 1 \leq j < i \leq m \quad (\text{B.458})$$

$$(\text{Row}_0(m), \langle \cdot, \cdot \rangle) : U_{ij}^{\text{Row}_0(m)} = \begin{cases} \frac{E_{ii} - E_{im} - E_{mi}}{\sqrt{3}}, & \text{if } 1 \leq i < m \\ \frac{E_{ij} + E_{ji} - E_{mi} - E_{im} - E_{mj} - E_{jm}}{\sqrt{6}}, & \text{if } 1 \leq j < i < m \end{cases} \quad (\text{B.459})$$

Putting the required diffeomorphisms and v_{ij}^g in Thm. 140 into Thm. 202 for ECM, LECM, OLM, and LSM, the corresponding FC layers can be readily obtained. $\square$

B.8.4 Proof of Thm. 143

Proof. First, we review the isometries between the open hemisphere and hyperboloid [195, Eqs. (4.1)–(4.2)], and the one between Poincaré ball and hyperboloid [182, Sec. 2.1]:

$$\psi_{\text{HS}^n \rightarrow \mathbb{H}^n} : (x_1, \dots, x_{n+1})^\top \in \text{HS}^n \mapsto \frac{1}{x_{n+1}} (1, x_1, \dots, x_n)^\top \in \mathbb{H}^n, \quad (\text{B.460})$$

$$\psi_{\mathbb{H}^n \rightarrow \mathbb{HS}^n} : (y_t, y_s^\top)^\top \in \mathbb{H}^n \mapsto \frac{1}{y_t} (y_s^\top, 1)^\top \in \mathbb{HS}^n, \quad (\text{B.461})$$

$$\psi_{\mathbb{H}^n \rightarrow \mathbb{P}^n} : (x_t, x_s^\top)^\top \in \mathbb{H}^n \mapsto \frac{x_s}{1 + x_t} \in \mathbb{P}^n, \quad (\text{B.462})$$

$$\psi_{\mathbb{P}^n \rightarrow \mathbb{H}^n} : y \in \mathbb{P}^n \mapsto \left(\frac{1 + \|y\|^2}{1 - \|y\|^2}, \frac{2y^\top}{1 - \|y\|^2} \right)^\top = \frac{1}{1 - \|y\|^2} \begin{pmatrix} 1 + \|y\|^2 \\ 2y \end{pmatrix} \in \mathbb{H}^n. \quad (\text{B.463})$$

For any $(x^\top, x_{n+1})^\top \in \mathbb{HS}^n$ and $y \in \mathbb{P}^n$, we have

$$\begin{aligned} \psi_{\mathbb{HS}^n \rightarrow \mathbb{P}^n} \left(\begin{pmatrix} x \\ x_{n+1} \end{pmatrix} \right) &= \psi_{\mathbb{H}^n \rightarrow \mathbb{P}^n} \circ \psi_{\mathbb{HS}^n \rightarrow \mathbb{H}^n} \left(\begin{pmatrix} x \\ x_{n+1} \end{pmatrix} \right) \\ &= \psi_{\mathbb{H}^n \rightarrow \mathbb{P}^n} \left(\frac{1}{x_{n+1}} \begin{pmatrix} 1 \\ x \end{pmatrix} \right) \\ &= \frac{x}{x_{n+1}} \frac{1}{1 + \frac{1}{x_{n+1}}} \\ &= \frac{x}{1 + x_{n+1}} \end{aligned} \quad (\text{B.464})$$

$$\begin{aligned} \psi_{\mathbb{P}^n \rightarrow \mathbb{HS}^n} (y) &= \psi_{\mathbb{H}^n \rightarrow \mathbb{HS}^n} \circ \psi_{\mathbb{P}^n \rightarrow \mathbb{H}^n} (y) \\ &= \psi_{\mathbb{H}^n \rightarrow \mathbb{HS}^n} \left(\frac{1}{1 - \|y\|^2} \begin{pmatrix} 1 + \|y\|^2 \\ 2y \end{pmatrix} \right) \\ &= \frac{1}{1 + \|y\|^2} \begin{pmatrix} 2y \\ 1 - \|y\|^2 \end{pmatrix} \end{aligned} \quad (\text{B.465})$$

□

B.8.5 Proof of Thm. 145

Proof. We denote $D = \mathcal{D}(H)$. By Sec. 2.9.2, we have

$$\begin{aligned} dY &= dD + dH \\ dD &= -\text{diag} \left((H^0)^{-1} \mathbb{D} (\exp_{*,Y}(dH)) \mathbf{1} \right). \end{aligned} \quad (\text{B.466})$$

Following Ionescu et al. [111], we denote the inner product $\langle \cdot, \cdot \rangle$ as $\cdot : \cdot$ for simplicity. By the invariance of differential and properties of trace [111, Eqs. 67–72], we have the

following:

$$\begin{aligned}
 \frac{\partial l}{\partial Y} : dY &= \frac{\partial l}{\partial Y} : dD + \frac{\partial l}{\partial Y} : dH \\
 &= \frac{\partial l}{\partial Y} : -\text{diag} \left((H^0)^{-1} \mathbb{D} \left(\exp_{*,Y}(dH) \right) \mathbf{1} \right) + \frac{\partial l}{\partial Y} : dH \\
 &\stackrel{(1)}{=} \text{tr} \left(-\text{Dv} \left(\frac{\partial l}{\partial Y} \right)^\top (H^0)^{-1} \mathbb{D} \left(\exp_{*,Y}(dH) \right) \mathbf{1} \right) + \frac{\partial l}{\partial Y} : dH \\
 &\stackrel{(2)}{=} \text{tr} \left(- \left[\mathbf{1} \text{Dv} \left(\frac{\partial l}{\partial Y} \right)^\top (H^0)^{-1} \right] \mathbb{D} \left(\exp_{*,Y}(dH) \right) \right) + \frac{\partial l}{\partial Y} : dH \\
 &= -(H^0)^{-1} \text{Dv} \left(\frac{\partial l}{\partial Y} \right) \mathbf{1}^\top : \mathbb{D} \left(\exp_{*,Y}(dH) \right) + \frac{\partial l}{\partial Y} : dH \\
 &= -\mathbb{D} \left((H^0)^{-1} \text{Dv} \left(\frac{\partial l}{\partial Y} \right) \mathbf{1}^\top \right) : \exp_{*,Y}(dH) + \frac{\partial l}{\partial Y} : dH \\
 &\stackrel{(3)}{=} \left[\frac{\partial l}{\partial Y} - \exp_{*,Y} \left(\mathbb{D} \left((H^0)^{-1} \text{Dv} \left(\frac{\partial l}{\partial Y} \right) \mathbf{1}^\top \right) \right) \right] : dH \\
 &\stackrel{(4)}{=} \text{off} \left[\frac{\partial l}{\partial Y} - \exp_{*,Y} \left(\mathbb{D} \left((H^0)^{-1} \text{Dv} \left(\frac{\partial l}{\partial Y} \right) \mathbf{1}^\top \right) \right) \right] : dH
 \end{aligned} \tag{B.467}$$

The above comes from the following.

(1)

$$A : \text{diag}(b) = \text{Dv}(A) : b, \quad \forall A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n, \tag{B.468}$$

$$a : b = a^\top b = \text{tr}(a^\top b), \quad \forall a, b \in \mathbb{R}^n. \tag{B.469}$$

(2) Cyclic property of the trace for matrices A, B , and C of compatible dimensions:
 $\text{tr}(ABC) = \text{tr}(CAB)$.

(3) For any $A \in \mathcal{S}^n$ and $S \in \mathcal{S}^n$, write $Y = U\Delta U^\top$ and let $L = L_{\text{exp}}$ be the Loewner matrix in Eq. (2.91). By the Daleckii–Krein formula in Eq. (2.90) and the properties of trace, we have

$$\begin{aligned}
 A : \exp_{*,Y}(S) &= A : U \left(L \otimes (U^\top S U) \right) U^\top \\
 &= U \left(L \otimes U^\top A U \right) U^\top : S \\
 &= \exp_{*,Y}(A) : S.
 \end{aligned} \tag{B.470}$$

(4) H has zero diagonal elements.

The invariance of the first-order differential gives

$$\frac{\partial l}{\partial Y} : dY = \frac{\partial l}{\partial H} : dH. \quad (\text{B.471})$$

By the last equation in Eq. (B.467), we can obtain $\frac{\partial l}{\partial H}$.

□

B.8.6 Proof of Thm. 146

Proof. Denoting by $f : \text{Cor}^+(n) \rightarrow \text{Row}_1^+(n)$ the map $f(C) = \mathcal{D}^*(C)C\mathcal{D}^*(C) = \Sigma$, we have

$$\text{Log}_{*,C}^* = \log_{*,\Sigma} \circ f_{*,C}. \quad (\text{B.472})$$

Combining with the differential of Log^* shown in Sec. 2.9.2, we have the following differential equation:

$$d\Sigma = \Delta dC \Delta - (V^0 \Sigma + \Sigma V^0), \quad (\text{B.473})$$

with $V^0 = \text{diag}((I_n + \Sigma)^{-1} \Delta dC \Delta \mathbf{1})$. Similarly to Thm. 145, we have the following:

$$\begin{aligned} \frac{\partial l}{\partial \Sigma} : d\Sigma &= \frac{\partial l}{\partial \Sigma} : (\Delta dC \Delta - (V^0 \Sigma + \Sigma V^0)) \\ &= \left(\Delta \frac{\partial l}{\partial \Sigma} \Delta \right) : dC - \frac{\partial l}{\partial \Sigma} : (V^0 \Sigma + \Sigma V^0) \\ &= \left(\Delta \frac{\partial l}{\partial \Sigma} \Delta \right) : dC - \left(\frac{\partial l}{\partial \Sigma} \Sigma + \Sigma \frac{\partial l}{\partial \Sigma} \right) : \text{diag}((I_n + \Sigma)^{-1} \Delta dC \Delta \mathbf{1}) \\ &= \left(\Delta \frac{\partial l}{\partial \Sigma} \Delta \right) : dC - \text{Dv} \left(\frac{\partial l}{\partial \Sigma} \Sigma + \Sigma \frac{\partial l}{\partial \Sigma} \right) : ((I_n + \Sigma)^{-1} \Delta dC \Delta \mathbf{1}) \\ &= \left(\Delta \frac{\partial l}{\partial \Sigma} \Delta \right) : dC - \text{tr}(\tilde{v}^\top (I_n + \Sigma)^{-1} \Delta dC \Delta \mathbf{1}) \\ &= \left(\Delta \frac{\partial l}{\partial \Sigma} \Delta \right) : dC - \text{tr}(\Delta \mathbf{1} \tilde{v}^\top (I_n + \Sigma)^{-1} \Delta dC) \\ &= \left(\Delta \frac{\partial l}{\partial \Sigma} \Delta \right) : dC - (\Delta (I_n + \Sigma)^{-1} \tilde{v} \mathbf{1}^\top \Delta) : dC \\ &= \left(\Delta \frac{\partial l}{\partial \Sigma} \Delta - \Delta (I_n + \Sigma)^{-1} \tilde{v} \mathbf{1}^\top \Delta \right) : dC \\ &= \left(\Delta \left(\frac{\partial l}{\partial \Sigma} - (I_n + \Sigma)^{-1} \tilde{v} \mathbf{1}^\top \right) \Delta \right) : dC. \end{aligned} \quad (\text{B.474})$$

By imposing symmetrization, we can obtain the results.

□

B.8.7 Proof of Thm. 144

As β -splitting is the inverse of β -concatenation [181], we only need to show the case w.r.t. β -concatenation. Besides, it suffices to prove the 2D case, which is shown in the following lemma.

Lemma 203. *Given $x_{ij} \in \mathbb{P}^{n_j}$ with $i \in \{1, \dots, N_i\}$ and $j \in \{1, \dots, N_j\}$, applying the β -concatenation sequentially 2 times in the order $j \rightarrow i$ is equivalent to a single β -concatenation along all indices simultaneously.*

Proof. Denoting $d = \sum_{j=1}^{N_j} n_j$ and $v_{ij} = \text{Log}_0(x_{ij})$, we have the following

$$\begin{aligned} & \text{Exp}_0 \left(\text{concat}_{i=1}^{N_i} \left(\beta_{N_i \times d} \beta_d^{-1} \text{concat}_{j=1}^{N_j} \left(\beta_d \beta_{n_j}^{-1} v_{ij} \right) \right) \right) \\ &= \text{Exp}_0 \left(\text{concat}_{i=1, j=1}^{i=N_i, j=N_j} \left(\beta_{N_i \times d} \beta_d^{-1} \beta_d \beta_{n_j}^{-1} v_{ij} \right) \right) \\ &= \text{Exp}_0 \left(\text{concat}_{i=1, j=1}^{i=N_i, j=N_j} \left(\beta_{N_i \times d} \beta_{n_j}^{-1} v_{ij} \right) \right). \end{aligned} \tag{B.475}$$

The last line implies the claim. □

A special case of the above lemma is where all n_j are identical.

Corollary 204. *Given $x_{ij} \in \mathbb{P}^n$ with $i \in \{1, \dots, N_i\}$ and $j \in \{1, \dots, N_j\}$, applying the β -concatenation sequentially 2 times in the order $j \rightarrow i$ is equivalent to a single β -concatenation along all indices simultaneously.*

Thm. 144 can be obtained by Thms. 203 and 204.

B.9 Adaptive Log-Euclidean Metrics

B.9.1 Proof of Thm. 147

Proof of Thm. 147. Let us first deal with (α, β) -LEM. Substituting the differential of the matrix logarithm into Thm. 33 directly yields the result.

Now, let us focus on LCM. Denote LCM, the standard Euclidean metric, and the metric on the Cholesky manifold [137] by g^{LC} , g^{E} , and g^{C} , respectively. By Tab. 2.5, $\{\mathcal{S}_{++}^n, g^{\text{LC}}\}$ is isometric to $\{\mathcal{L}_{++}^n, g^{\text{C}}\}$, with the Cholesky decomposition Chol as an isometry. This is exactly how Lin [137] derived LCM. So, the key point lies in the Cholesky metric g^{C} . Let us reveal why it is defined in this way. In fact, g^{C} is derived

from g^E by φ_{ln} . Simple computations show that

$$(\varphi_{ln})_{*,L}(V) = \lfloor V \rfloor + \mathbb{D}(L)^{-1}\mathbb{D}(V), \quad (\text{B.476})$$

where $V \in T_L \mathcal{L}_{++}^n$. By Eq. (B.476), Tab. 2.5 can be rewritten as

$$g_L^C(X, Y) = g^E \left((\varphi_{ln})_{*,L}(X), (\varphi_{ln})_{*,L}(Y) \right). \quad (\text{B.477})$$

Therefore, $\varphi_{ln} : \mathcal{L}_{++}^n \rightarrow \text{LT}^n$ is an isometry. By transitivity, $\psi_{\text{LC}} : \mathcal{S}_{++}^n \rightarrow \text{LT}^n$ is also an isometry. $\square$

B.9.2 Proof of Thm. 148

Proof of Thm. 148. As $\mathbb{R}^{n(n+1)/2} \cong \text{LT}^n \cong \mathcal{S}^n$, LCM is therefore a pullback metric from the standard Euclidean space $\mathcal{S}^n$. Second, any two Euclidean spaces of the same finite dimension are naturally isometric; hence, (α, β) -LEM is also a pullback metric from the standard Euclidean space $\mathcal{S}^n$. $\square$

B.9.3 Proof of Thm. 149

Proof of Thm. 149. By the definitions in Eqs. (6.2) to (6.5), the Hilbert-space and isomorphism claims follow directly. It remains to establish the geometric claims. As every Euclidean space is an abelian Lie group, $\{\mathcal{S}_{++}^n, \odot_\phi\}$ is an abelian Lie group. The geodesic distance in Eq. (6.6) also follows immediately because ϕ is a Riemannian isometry.

We only need to prove Eqs. (6.7) to (6.9). Note that in the Euclidean space $\mathcal{S}^n$, for any $x, y \in \mathcal{S}^n$ and tangent vector $v \in T_x \mathcal{S}^n \cong \mathcal{S}^n$, we have the following:

$$\text{Exp}_x v = x + v, \quad (\text{B.478})$$

$$\text{Log}_x y = y - x, \quad (\text{B.479})$$

$$\text{PT}_{x \rightarrow y} v = v. \quad (\text{B.480})$$

By the isometry of ϕ , we can readily obtain Eqs. (6.7) to (6.9). $\square$

B.9.4 Proof of Thm. 150

Proof of Thm. 150. Obviously, $\log_\alpha^{-1}$ is the inverse of $\log_\alpha$. What follows is to verify the smoothness of $\log_\alpha$ and its inverse.

According to Magnus and Neudecker [144, Thm. 8.9], the map producing an eigenvalue or an eigenvector from a real symmetric matrix is C^∞ . Recalling $\log_\alpha$ and its inverse map $\log_\alpha^{-1}$, it is obvious that they comprise arithmetic calculations or compositions of smooth maps. Therefore, $\log_\alpha$ is a diffeomorphism with inverse $\log_\alpha^{-1}$. $\square$

B.9.5 Proof of Thm. 152

Proof of Thm. 152. This is a direct result of Thm. 149. $\square$

B.9.6 Proof of Thm. 154

Proof of Thm. 154. The differentials of $\log_\alpha^{-1}$ and $\log_\alpha$ can be derived similarly. In the following, we only present the process of deriving the differential of $\log_\alpha$.

First, let us recall the differentials of eigenvalues and eigenvectors. Magnus and Neudecker [144, Thm. 8.9] offers their Euclidean differentials, which are the exact formulations for differentials under the canonical base on SPD manifolds. Thus, we can readily obtain the differentials of eigenvalues and eigenvectors as follows:

$$\sigma_{*,S}(V) = u^\top V u, \quad (\text{B.481})$$

$$u_{*,S}(V) = (\sigma I_n - S)^+ V u, \quad (\text{B.482})$$

where $Su = \sigma u$, $u^\top u = 1$, and $(\cdot)^+$ is the Moore–Penrose inverse.

By the RHS of Eq. (6.30), the differential map of $\log_\alpha$ is

$$\begin{aligned} & (\log_\alpha)_{*,S}(V) \\ &= U_{*,S}(V) \log_\alpha(\Sigma) U^\top + U (\log_\alpha)_{*,\Sigma}(\Sigma_{*,S}(V)) U^\top \\ &+ U \log_\alpha(\Sigma) U_{*,S}^\top(V) \\ &= Q + Q^\top + U (\log_\alpha)_{*,\Sigma}(\Sigma_{*,S}(V)) U^\top, \end{aligned} \quad (\text{B.483})$$

where $Q = U_{*,S}(V) \log_\alpha(\Sigma) U^\top$.

For the differential of diagonal logarithm, it is

$$(\log_\alpha)_{*,\Sigma}(\Sigma_{*,S}(V)) = A \frac{1}{\Sigma} \Sigma_{*,S}(V), \quad (\text{B.484})$$

where A is defined in Eq. (6.31).

Denote the eigenvectors and eigenvalues of $S = U \Sigma U^\top$ by $U = (u_1, \dots, u_n)$ and $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_n)$. By Eqs. (B.481) to (B.484), the differential of $\log_\alpha$ can be

obtained. $\square$

B.9.7 Proof of Thm. 155

Proof of Thm. 155. Following the notation in the proposition, we prove the result as follows. By abuse of notation, in the following, we omit the wide tilde $\sim$.

Now, we proceed to deal with the differential of $\log_\alpha^{-1}$. We rewrite the formula of $\log_\alpha^{-1}$ as

$$\log_\alpha^{-1}(X) \tag{B.485}$$

$$= U \operatorname{diag}(a_1^{\sigma_1}, \dots, a_n^{\sigma_n}) U^\top, \tag{B.486}$$

$$= U \operatorname{diag}(e^{\log(a_1)\sigma_1}, \dots, e^{\log(a_n)\sigma_n}) U^\top, \tag{B.487}$$

$$= U \operatorname{diag}\left(\sum_{k=0}^{\infty} \frac{(\log(a_1)\sigma_1)^k}{k!}, \dots, \sum_{k=0}^{\infty} \frac{(\log(a_n)\sigma_n)^k}{k!}\right) U^\top, \tag{B.488}$$

$$= U \left(\sum_{k=0}^{\infty} \frac{(B\Sigma)^k}{k!}\right) U^\top, \tag{B.489}$$

$$= \sum_{k=0}^{\infty} \frac{(PX)^k}{k!} \tag{B.490}$$

where $P = UBU^\top$, U is obtained from the eigendecomposition $X = U\Sigma U^\top$, and $B = \operatorname{diag}(\log(a_1), \dots, \log(a_n))$ is diagonal. By the properties of normed vector algebras [197, Prop. 15.14], we can obtain the last equation. Then, we can compute the differential of $\log_\alpha^{-1}$ by curves. Given a curve c on $\mathcal{S}^n$ starting at X with initial velocity $V \in T_X \mathcal{S}^n$, write $c(t) = U(t)\Sigma(t)U(t)^\top$ and define $P(t) = U(t)BU(t)^\top$, so that $P(0) = P$. We have

$$\begin{aligned} (\log_\alpha^{-1})_{*,X}(V) &= \left. \frac{d}{dt} \right|_{t=0} \log_\alpha^{-1}(c(t)) \\ &= \left. \frac{d}{dt} \right|_{t=0} \sum_{k=0}^{\infty} \frac{(P(t)c(t))^k}{k!}. \end{aligned} \tag{B.491}$$

Term-by-term differentiation gives

$$\begin{aligned} &(\log_\alpha^{-1})_{*,X}(V) \\ &= \sum_{k=1}^{\infty} \frac{1}{k!} \left(\sum_{l=0}^{k-1} (PX)^{k-l-1} \left. \frac{d}{dt} \right|_{t=0} (P(t)c(t))(PX)^l \right). \end{aligned} \tag{B.492}$$

By the chain rule, we have

$$\left. \frac{d}{dt} \right|_{t=0} (P(t)c(t)) = P'(0)X + PV. \quad (\text{B.493})$$

$P'(0)$ is obtained by

$$\begin{aligned} P'(0) &= \left. \frac{d}{dt} \right|_{t=0} (U(t)BU(t)^\top) \\ &= U'(0)BU^\top + UBU'(0)^\top \\ &= D_UBU^\top + UBD_U^\top, \end{aligned} \quad (\text{B.494})$$

where D_U is derived from the differential of eigenvectors,

$$D_U = ((\sigma_1 I_n - X)^+ V u_1 \quad \cdots \quad (\sigma_n I_n - X)^+ V u_n). \quad (\text{B.495})$$

Substituting Eqs. (B.493) to (B.495) into Eq. (B.492) yields the differential of $\log_\alpha^{-1}$. $\square$

B.9.8 Proof of Thm. 156

Proof of Thm. 156. Obviously, the metric space $\{\mathcal{S}_{++}^n, d^{\text{ALE}}\}$ is isometric to the space $\mathcal{S}^n$ endowed with the standard Euclidean distance. Therefore, the weighted Fréchet mean of $\{S_i\}$ in $\mathcal{S}_{++}^n$ corresponds to the weighted Fréchet mean of associated points $\{\log_\alpha(S_i)\}$ in $\mathcal{S}^n$. The weighted Fréchet means in Euclidean spaces are clearly the familiar weighted means. $\square$

B.9.9 Proof of Thm. 157

Proof of Thm. 157. As $\log_\alpha$ is a Riemannian isometry and $\mathcal{S}^n$ is bi-invariant, ALEM is therefore bi-invariant. $\square$

B.9.10 Proof of Thm. 158

Proof of Thm. 158. Following the notation in this proposition, we proceed as follows. The right-hand side can be rewritten as

$$\begin{aligned}
 (\text{FM}(S_1^\beta, \dots, S_m^\beta)) &= \log_\alpha^{-1} \left(\sum_{i=1}^m \frac{1}{m} \beta \log_\alpha(S_i) \right) \\
 &= \log_\alpha^{-1} \left(\beta \sum_{i=1}^m \frac{1}{m} \log_\alpha(S_i) \right) \\
 &= \left[\log_\alpha^{-1} \left(\sum_{i=1}^m \frac{1}{m} \log_\alpha(S_i) \right) \right]^\beta \\
 &= (\text{FM}(S_1, \dots, S_m))^\beta.
 \end{aligned} \tag{B.496}$$

□

B.9.11 Proof of Thm. 159

Proof of Thm. 159. Recalling Eq. (6.24), Properties U1 and U2 obviously hold.

When the SPD matrices $\{A_i\}_{i \leq n}$ commute, we have

$$\text{FM}(\{A_i\}) = \left(\prod_i A_i \right)^{\frac{1}{n}}. \tag{B.497}$$

With Eq. (B.497), Properties V1–V4 can be easily proved.

□

B.9.12 Proof of Thm. 160

Proof of Thm. 160. Obviously, for a given SPD matrix S ,

$$\log_\alpha(RSR^\top) = R \log_\alpha(S) R^\top, \tag{B.498}$$

$$\log_\alpha(s^2 S) = U \left(\log_\alpha(s^2 I_n) + \log_\alpha(\Sigma) \right) U^\top, \tag{B.499}$$

where $S = U \Sigma U^\top$ is the eigendecomposition. These identities yield the result.

□

B.9.13 Proof of Thm. 161

Proof of Thm. 161. The three equations can be directly obtained.

□

B.9.14 Proof of Thm. 163

Proof of Thm. 163. The input gradient follows from the Daleckii–Krein formula reviewed in Eqs. (2.90) and (2.91). Now, let us focus on the gradient with respect to A . Differentiating both sides of Eq. (6.31) gives

$$dX = (*) + U(dA \otimes \log(\Sigma))U^\top, \quad (\text{B.500})$$

where $(*)$ denotes other terms involving dU and $d\Sigma$. According to the invariance of the first-order differential form, we have

$$\nabla_X L : dX \quad (\text{B.501})$$

$$= \nabla_S L : dS + \nabla_X L : U(dA \otimes \log(\Sigma))U^\top \quad (\text{B.502})$$

$$= \nabla_S L : dS + ([U^\top(\nabla_X L)U] \otimes \log(\Sigma)) : dA, \quad (\text{B.503})$$

where $A : B = \text{tr}(A^\top B)$ is the Euclidean Frobenius inner product. From the second term on the RHS of Eq. (B.503), we can obtain the gradient with respect to A . $\square$

B.9.15 Proof of Thm. 164

Proof of Thm. 164. The derivation follows the same logic as Thm. 163. We only need to show the derivation of Eq. (6.35). Similarly to Thm. 163, we have the following:

$$dX = (*) + U \left[dA \otimes \left(\text{diag}(a_1^{\Sigma_{11}}, \dots, a_n^{\Sigma_{nn}}) \frac{-\Sigma}{A^2} \right) \right] U^\top, \quad (\text{B.504})$$

$$\nabla_X L : dX \quad (\text{B.505})$$

$$= \nabla_S L : dS + \left\{ [U^\top(\nabla_X L)U] \otimes \left(\text{diag}(a_1^{\Sigma_{11}}, \dots, a_n^{\Sigma_{nn}}) \frac{-\Sigma}{A^2} \right) \right\} : dA. \quad (\text{B.506})$$

$\square$

B.9.16 Proof of Thm. 167

Proof of Thm. 167. Following Nguyen [157], Nguyen and Yang [159], we first define gyrostructures under ALEM:

$$P \oplus^{\text{ALE}} Q = \text{Exp}_P(\text{PT}_{I_n \rightarrow P}(\text{Log}_{I_n}(Q))), \quad (\text{B.507})$$

$$\text{gyr}[P, Q]R = (\ominus(P \oplus^{\text{ALE}} Q)) \oplus^{\text{ALE}} (P \oplus^{\text{ALE}} (Q \oplus^{\text{ALE}} R)), \quad (\text{B.508})$$

$$t \odot^{\text{ALE}} P = \text{Exp}_{I_n}(t \text{Log}_{I_n}(P)), \quad (\text{B.509})$$

$$\ominus P = -1 \odot^{\text{ALE}} P = \text{Exp}_{I_n}(-\text{Log}_{I_n}(P)), \quad (\text{B.510})$$

$$\langle P, Q \rangle_{\text{gyr}} = \langle \text{Log}_{I_n}(P), \text{Log}_{I_n}(Q) \rangle_{I_n}, \quad (\text{B.511})$$

$$\|P\|_{\text{gyr}} = \sqrt{\langle P, P \rangle_{\text{gyr}}}, \quad (\text{B.512})$$

$$d_{\text{gyr}}(P, Q) = \|\ominus P \oplus^{\text{ALE}} Q\|_{\text{gyr}}, \quad (\text{B.513})$$

where $P, Q, R \in \mathcal{S}_{++}^n$, and I_n is the identity matrix. The above operations are called gyroaddition, gyroautomorphism, scalar gyromultiplication, gyroinverse, gyroinner product, gyronorm, and gyrodistance. Simple computations show that Eq. (B.507) and Eq. (B.509) are exactly $\oplus^{\text{ALE}}$ and $\odot^{\text{ALE}}$ in Thm. 152. As indicated by Thm. 152, $\{\mathcal{S}_{++}^n, \oplus^{\text{ALE}}, \odot^{\text{ALE}}\}$ forms a gyrovector space [158, Def. 1]. In the following proof, we use $\odot^{\text{ALE}}$ and $\oplus^{\text{ALE}}$.

The gyro MLR [159] under ALEM is defined as

$$\begin{aligned} p(y = k \mid S) \\ \propto \exp \left(\text{sign}(\langle \tilde{A}_k, \text{Log}_{P_k}(S) \rangle_{P_k}) \|\tilde{A}_k\|_{P_k} \bar{d}(S, H_{\tilde{A}_k, P_k}) \right), \end{aligned} \quad (\text{B.514})$$

where $P_k \in \mathcal{S}_{++}^n$ and $\tilde{A}_k \in T_{P_k} \mathcal{S}_{++}^n$. $\bar{d}(S, H_{\tilde{A}_k, P_k})$ is the margin distance to the SPD hyperplane $H_{\tilde{A}_k, P_k}$, which is defined as

$$\bar{d}(S, H_{\tilde{A}_k, P_k}) = \sin(\angle S P_k Q^*) d_{\text{gyr}}(S, P_k), \quad (\text{B.515})$$

$$Q^* = \underset{Q \in H_{\tilde{A}_k, P_k} \setminus \{P_k\}}{\text{argmax}} (\cos(\angle S P_k Q)), \quad (\text{B.516})$$

$$\cos(\angle S P_k Q) = \frac{\langle \ominus P_k \oplus^{\text{ALE}} Q, \ominus P_k \oplus^{\text{ALE}} S \rangle_{\text{gyr}}}{\|\ominus P_k \oplus^{\text{ALE}} Q\|_{\text{gyr}} \|\ominus P_k \oplus^{\text{ALE}} S\|_{\text{gyr}}}, \quad (\text{B.517})$$

$$H_{\tilde{A}_k, P_k} = \{S \in \mathcal{S}_{++}^n \mid \langle \text{Log}_{P_k} S, \tilde{A}_k \rangle_{P_k} = 0\}. \quad (\text{B.518})$$

Eqs. (B.515), (B.517) and (B.518) are called the SPD pseudo-gyrodistance, SPD gyrocosine, and SPD gyrohyperplane.

For simplicity, we further omit the subscript k in P_k and $\tilde{A}_k$. Eq. (B.518) can be

simplified:

$$\begin{aligned}
 & \langle \text{Log}_P S, \tilde{A} \rangle_P \\
 & \stackrel{(1)}{=} \left\langle (\log_\alpha^{-1})_{*, \log_\alpha(P)} (\log_\alpha(S) - \log_\alpha(P)), \tilde{A} \right\rangle_P \\
 & \stackrel{(2)}{=} \left\langle (\log_\alpha)_{*, P} \circ (\log_\alpha^{-1})_{*, \log_\alpha(P)} (\log_\alpha(S) - \log_\alpha(P)), (\log_\alpha)_{*, P} (\tilde{A}) \right\rangle \\
 & = \left\langle \log_\alpha(S) - \log_\alpha(P), (\log_\alpha)_{*, P} (\tilde{A}) \right\rangle.
 \end{aligned} \tag{B.519}$$

The above derivation comes from the following.

(1) Eq. (6.18).

(2) The definition of ALEM.

Similarly, a simple computation shows that Eq. (B.517) can also be simplified as

$$\frac{\langle -\log_\alpha(P) + \log_\alpha(Q), -\log_\alpha(P) + \log_\alpha(S) \rangle}{\|-\log_\alpha(P) + \log_\alpha(Q)\|_F \|-\log_\alpha(P) + \log_\alpha(S)\|_F}. \tag{B.520}$$

Together with Eqs. (B.519) and (B.520), Eq. (B.515) is equivalent to the distance to the hyperplane in the Euclidean space. Therefore, Eq. (B.515) has a closed-form solution:

$$\begin{aligned}
 & \bar{d}(S, H_{\tilde{A}, P}) \\
 & = \frac{|\langle \log_\alpha(S) - \log_\alpha(P), \bar{A} \rangle|}{\|\bar{A}\|_F} \\
 & = \frac{|\langle \log_\alpha(S) - \log_\alpha(P), \bar{A} \rangle|}{\|\tilde{A}\|_P},
 \end{aligned} \tag{B.521}$$

where $\bar{A} = (\log_\alpha)_{*, P} (\tilde{A})$. Substituting Eq. (B.521) into Eq. (B.514) yields the claimed result. $\square$

B.9.17 Proof of Thm. 165

Proof of Thm. 165. To derive the ALEM-specific positive-scalar update, consider the general RSGD update reviewed in Sec. 2.7. For a minimization parameter w on an n -dimensional smooth connected Riemannian manifold $\mathcal{M}$, we have

$$w^{(t+1)} = \text{Exp}_{w^{(t)}}(-\gamma^{(t)} \pi_{w^{(t)}}(\nabla_{w^{(t)}} L)), \tag{B.522}$$

where $\text{Exp}_w(\cdot) : T_w\mathcal{M} \rightarrow \mathcal{M}$ is the Riemannian exponential map, which maps a tangent vector at w back into the manifold $\mathcal{M}$, and $\pi_w(\cdot) : \mathbb{R}^n \rightarrow T_w\mathcal{M}$ is the projection operator, projecting an ambient Euclidean vector into the tangent space at w . In the case of the SPD manifold, for all $S \in \mathcal{S}_{++}^n$, $X \in \mathbb{R}^{n \times n}$, and $V \in \mathcal{S}^n$, the exponential map and projection operator are formulated as follows:

$$\pi_S(X) = S \frac{X + X^\top}{2} S, \quad (\text{B.523})$$

$$\text{Exp}_S(V) = S^{1/2} \exp(S^{-1/2} V S^{-1/2}) S^{1/2}, \quad (\text{B.524})$$

where $\exp(\cdot)$ is the matrix exponential. For more details about Eq. (B.523) and Eq. (B.524), see Yger [225] and Amari [3]. Substituting Eq. (B.523) and Eq. (B.524) into Eq. (B.522) immediately yields Eq. (6.36). $\square$

B.9.18 Proof of Thm. 166

Proof of Thm. 166. Without loss of generality, we focus on the equivalence between $b = B_{11}$ and $a = a_1$. Note that b is essentially expressed as $b = \log(a)$. Suppose $b^{(t)} = \log(a^{(t)})$. Then, we have

$$\begin{aligned} \nabla_{a^{(t)}} L &= \nabla_{b^{(t)}} L \left. \frac{\partial \log(a)}{\partial a} \right|_{a^{(t)}} \\ &= \nabla_{b^{(t)}} L \frac{1}{a^{(t)}}. \end{aligned} \quad (\text{B.525})$$

By Eq. (6.36), $\log(a^{(t+1)})$ is

$$\begin{aligned} \log(a^{(t+1)}) &= \log\left(a^{(t)} e^{-\gamma^{(t)} a^{(t)} \nabla_{a^{(t)}} L}\right) \\ &= \log(a^{(t)}) - \gamma^{(t)} a^{(t)} \nabla_{a^{(t)}} L \\ &= \log(a^{(t)}) - \gamma^{(t)} a^{(t)} (\nabla_{b^{(t)}} L / a^{(t)}) \\ &= \log(a^{(t)}) - \gamma^{(t)} \nabla_{b^{(t)}} L \\ &= b^{(t)} - \gamma^{(t)} \nabla_{b^{(t)}} L. \end{aligned} \quad (\text{B.526})$$

The last row is the ESGD update formula for b .

Therefore, if $b^{(0)} = \log(a^{(0)})$, the two optimization procedures yield equivalent iterates throughout training. $\square$

B.10 Product Cholesky Metrics

B.10.1 Proof of Thm. 169

Proof. Since θ -DPM is the product metric of $\{\text{LT}^0(n), g^E\}$ and n copies of $\{\mathbb{R}_{++}, g^{\theta-E}\}$, we first show the Riemannian operators on $\{\mathbb{R}_{++}, g^{\theta-E}\}$. We can then readily obtain the Riemannian operators on $\{\mathcal{L}_{++}^n, g^{\theta-DE}\}$ by the principles of product metrics.

As shown by Thanwerdas and Pennec [194], $g^{\theta-E}$ is the pullback metric of g^E by the power function $P_\theta(\cdot)$ and scaled by $\frac{1}{\theta^2}$, expressed as $g^{\theta-E} = \frac{1}{\theta^2} P_\theta^* g^E$. Besides, as constant scaling does not change the Christoffel symbols, the geodesic, Riemannian logarithm and exponential maps, and parallel transport along a geodesic remain the same under $g^{\theta-E}$ and $P_\theta^* g^E$. These Riemannian operators under $P_\theta^* g^E$ can be obtained by the properties of Riemannian isometries (Thm. 33). Specifically, given $p, q \in \mathbb{R}_{++}$ and $w, v \in T_p \mathbb{R}_{++}$, we have the following:

$$(P_\theta)_{*,p}(v) = \theta p^{\theta-1} v, \quad (\text{B.527})$$

$$g_p^{\theta-E}(v, w) = \frac{1}{\theta^2} g^E\left((P_\theta)_{*,p}(v), (P_\theta)_{*,p}(w)\right) = \langle p^{\theta-1} v, p^{\theta-1} w \rangle = p^{2(\theta-1)} v w, \quad (\text{B.528})$$

$$\gamma_{(p,v)}(t) = P_\theta^{-1}\left(P_\theta(p) + t(P_\theta)_{*,p}(v)\right) \quad (\text{B.529})$$

$$= (p^\theta + t\theta p^{\theta-1} v)^{\frac{1}{\theta}} \quad (\text{B.530})$$

$$= p(1 + t\theta p^{-1} v)^{\frac{1}{\theta}}, \text{ with } t \in \{t \in \mathbb{R} \mid 1 + t\theta p^{-1} v \in \mathbb{R}_{++}\}, \quad (\text{B.531})$$

$$\text{Log}_p(q) = (P_\theta)_{*,p}^{-1}(P_\theta(q) - P_\theta(p)) \quad (\text{B.532})$$

$$= \frac{1}{\theta} p^{1-\theta} (q^\theta - p^\theta) = \frac{1}{\theta} p \left(\left(\frac{q}{p} \right)^\theta - 1 \right), \quad (\text{B.533})$$

$$\text{PT}_{p \rightarrow q}(v) = (P_\theta)_{*,q}^{-1}\left((P_\theta)_{*,p}(v)\right) = \left(\frac{q}{p} \right)^{1-\theta} v. \quad (\text{B.534})$$

The geodesic distance between p and q under $g^{\theta-E}$ is given by

$$d^2(p, q) = g_p^{\theta-E}(\text{Log}_p(q), \text{Log}_p(q)) = \frac{1}{\theta^2} (q^\theta - p^\theta)^2. \quad (\text{B.535})$$

The weighted Fréchet mean (WFM) of $\{p_i \in \mathbb{R}_{++}\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$ satisfying $w_i > 0$ for all i and $\sum_i w_i = 1$ under $g^{\theta-E}$ is defined as

$$\text{WFM}(\{w_i\}, \{p_i\}) = \underset{p \in \mathbb{R}_{++}}{\text{argmin}} \sum_{i=1}^N w_i d^2(p, p_i). \quad (\text{B.536})$$

Obviously, the WFM of $\{p_i\}$ under $g^{\theta-E}$ is the same as the one under $P_\theta^* g^E$. Due to the isometry of $P_\theta^* g^E$ to g^E , the WFM of $\{p_i\}$ under $g^{\theta-E}$ can be calculated as

$$\text{WFM}(\{w_i\}, \{p_i\}) = P_\theta^{-1}(\text{WFM}^E(\{w_i\}, \{P_\theta(p_i)\})) = \left(\sum_{i=1}^N w_i p_i^\theta\right)^{\frac{1}{\theta}}, \quad (\text{B.537})$$

where WFM^E in Eq. (B.537) is the Euclidean WFM, which is the familiar weighted average.

So far, we have obtained all the necessary Riemannian operators on $\{\mathbb{R}_{++}, g^{\theta-E}\}$. Combining these results with the Euclidean space $\text{LT}^0(n)$ yields the results in the theorem. $\square$

B.10.2 Proof of Thm. 170

Proof. As in the proof of Thm. 169, we only need to show the Riemannian operators on $\{\mathbb{R}_{++}, g^{m\text{-BW}}\}$ with $m \in \mathbb{R}_{++}$. Expressions for the Riemannian operators under GBWM can be found in Han et al. [93]. Here, we further simplify the associated expressions for the one-dimensional case. Specifically, given $p, q \in \mathbb{R}_{++}$ and $w, v \in T_p \mathbb{R}_{++}$, we have the following:

$$\mathcal{L}_{p,m}(v) = \frac{v}{2mp}, \quad (\text{B.538})$$

$$g_p^{m\text{-BW}}(v, w) = \frac{1}{2} \langle \mathcal{L}_{p,m}(v), w \rangle = \frac{vw}{4mp}, \quad (\text{B.539})$$

$$\gamma_{(p,v)}(t) = p + tv + \mathcal{L}_{p,m}(tv)^2 m^2 p = p + tv + \frac{(tv)^2}{4p} = p \left(1 + \frac{t}{2} \frac{v}{p}\right)^2, \quad (\text{B.540})$$

$$\text{Log}_p(q) = 2 \left(m(m^{-2}pq)^{\frac{1}{2}} - p\right) = 2 \left((pq)^{\frac{1}{2}} - p\right) = 2p \left(\left(\frac{q}{p}\right)^{\frac{1}{2}} - 1\right), \quad (\text{B.541})$$

$$\begin{aligned} d^2(p, q) &= g_p^{m\text{-BW}}(\text{Log}_p(q), \text{Log}_p(q)) = \frac{1}{4mp} 4 \left((pq)^{\frac{1}{2}} - p\right)^2 \\ &= \frac{1}{mp} \left(pq - 2p^{\frac{3}{2}}q^{\frac{1}{2}} + p^2\right) \\ &= m^{-1} \left(q - 2p^{\frac{1}{2}}q^{\frac{1}{2}} + p\right) \\ &= \left(m^{-\frac{1}{2}} \left(q^{\frac{1}{2}} - p^{\frac{1}{2}}\right)\right)^2. \end{aligned} \quad (\text{B.542})$$

As shown by Han et al. [93], GBWM on $\mathcal{S}_{++}^n$ is the pullback metric of BWM by $\pi(S) = M^{-\frac{1}{2}} S M^{-\frac{1}{2}}$ for all $S \in \mathcal{S}_{++}^n$ with $M \in \mathcal{S}_{++}^n$. For the specific $\mathbb{R}_{++} \cong \mathcal{S}_{++}^1$, the

isometry is simplified as

$$\pi(p) = m^{-1}p, \forall p \in \mathbb{R}_{++} \text{ with } m \in \mathbb{R}_{++}. \quad (\text{B.543})$$

The geodesic $\tilde{\gamma}_{(p,v)}(t)$ under BWB on $\mathbb{R}_{++}$ exists in the interval $\{t \in \mathbb{R} \mid 1 + t\mathcal{L}_p(v) \in \mathbb{R}_{++}\}$ [145], which can be simplified as $\left\{t \in \mathbb{R} \mid 1 + t\frac{v}{2p} \in \mathbb{R}_{++}\right\}$. Therefore, the geodesic $\gamma_{(p,v)}(t)$ under $g^{m\text{-BW}}$ exists in the interval:

$$\left\{t \in \mathbb{R} \mid 1 + t\frac{\pi_{*,p}(v)}{2\pi(p)} \in \mathbb{R}_{++}\right\} = \left\{t \in \mathbb{R} \mid 1 + t\frac{v}{2p} \in \mathbb{R}_{++}\right\}. \quad (\text{B.544})$$

For BWB, the parallel transport is [196, Tab. 6]

$$\widetilde{\text{PT}}_{p \rightarrow q}(v) = \left(\frac{q}{p}\right)^{\frac{1}{2}} v. \quad (\text{B.545})$$

Therefore, the parallel transport on $\{\mathbb{R}_{++}, g^{m\text{-BW}}\}$ is

$$\begin{aligned} \text{PT}_{p \rightarrow q}(v) &= \pi_{*,q}^{-1} \left(\widetilde{\text{PT}}_{\pi(p) \rightarrow \pi(q)}(\pi_{*,p}(v)) \right) = \pi_{*,q}^{-1} \left(\left(\frac{\pi(q)}{\pi(p)} \right)^{\frac{1}{2}} \pi_{*,p}(v) \right) \\ &= \left(\frac{q}{p} \right)^{\frac{1}{2}} v. \end{aligned} \quad (\text{B.546})$$

Finally, we show the WFM on $\{\mathbb{R}_{++}, g^{m\text{-BW}}\}$. Given $\{p_i \in \mathbb{R}_{++}\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$ satisfying $w_i > 0$ for all i and $\sum_{i=1}^N w_i = 1$, the WFM on $\{\mathbb{R}_{++}, g^{m\text{-BW}}\}$ is

$$\begin{aligned} \text{WFM}(\{w_i\}, \{p_i\}) &= \underset{p \in \mathbb{R}_{++}}{\text{argmin}} \sum_{i=1}^N w_i d^2(p, p_i) \\ &= \underset{p \in \mathbb{R}_{++}}{\text{argmin}} \sum_{i=1}^N w_i m^{-1} \left(p^{\frac{1}{2}} - p_i^{\frac{1}{2}} \right)^2 \\ &= \underset{p \in \mathbb{R}_{++}}{\text{argmin}} \sum_{i=1}^N w_i \left(p^{\frac{1}{2}} - p_i^{\frac{1}{2}} \right)^2 \\ &= \underset{p \in \mathbb{R}_{++}}{\text{argmin}} \sum_{i=1}^N w_i \left(p - 2p^{\frac{1}{2}} p_i^{\frac{1}{2}} \right) \\ &= \underset{p \in \mathbb{R}_{++}}{\text{argmin}} p - 2p^{\frac{1}{2}} \sum_{i=1}^N w_i p_i^{\frac{1}{2}}. \end{aligned} \quad (\text{B.547})$$

Let $f(p) = p - 2p^{\frac{1}{2}} \sum_{i=1}^N w_i p_i^{\frac{1}{2}}$. Then, the first- and second-order derivatives are

$$\frac{df}{dp} = 1 - p^{-\frac{1}{2}} \sum_i w_i p_i^{\frac{1}{2}}, \quad (\text{B.548})$$

$$\frac{d^2 f}{dp^2} = \frac{1}{2} p^{-\frac{3}{2}} \sum_i w_i p_i^{\frac{1}{2}} > 0, \quad \forall p \in \mathbb{R}_{++}. \quad (\text{B.549})$$

Therefore, the optimal solution can be obtained by setting Eq. (B.548) equal to 0:

$$\frac{df}{dp} = 0 \Rightarrow p^* = \left(\sum_i w_i p_i^{\frac{1}{2}} \right)^2. \quad (\text{B.550})$$

Combining the above results with the Euclidean geometry on $\text{LT}^0(n)$, one can readily obtain the results. $\square$

B.10.3 Proof of Thm. 172

Proof. The differential of DPow_θ at $\mathbb{L} \in \text{Diag}^+(n)$ is

$$(\text{DPow}_\theta)_{*,\mathbb{L}}(\mathbb{V}) = \theta \mathbb{L}^{\theta-1} \mathbb{V}, \quad \forall \mathbb{V} \in T_{\mathbb{L}} \text{Diag}^+(n). \quad (\text{B.551})$$

Substituting Eq. (B.551) into Thm. 171 yields the result. $\square$

B.10.4 Proof of Thm. 173

We first present a useful lemma.

Lemma 205. *The Riemannian exponential and logarithmic maps and parallel transport along the geodesic are the same under $(\theta, \mathbb{M})$ -DBWM and $\theta/2$ -DPM.*

Proof. This follows directly from Thms. 169 and 170. $\square$

Now we begin to prove Thm. 173.

Proof. We first derive the expressions for DPM with a generic nonzero parameter θ . According to Thm. 205, the expressions for $(\theta, \mathbb{M})$ -DBWM then follow by replacing θ with $\theta/2$. We omit the superscript $\mathcal{C}$ for simplicity.

For $X \in T_{I_n} \mathcal{L}_{++}^n$, we have the following:

$$\text{Log}_{I_n}(L) = \lfloor L \rfloor + \frac{1}{\theta} (\mathbb{L}^\theta - I_n), \quad (\text{B.552})$$

$$\text{PT}_{I_n \rightarrow L}(X) = \lfloor X \rfloor + \mathbb{L}^{1-\theta} \mathbb{X}, \quad (\text{B.553})$$

$$\text{Exp}_{I_n} X = \lfloor X \rfloor + (I_n + \theta \mathbb{X})^{\frac{1}{\theta}}. \quad (\text{B.554})$$

For the binary operation, substituting Eqs. (B.552) and (B.553) into Eq. (2.77) gives

$$\begin{aligned} L \oplus K &= \text{Exp}_L (\text{PT}_{I_n \rightarrow L} (\text{Log}_{I_n}(K))) \\ &= \text{Exp}_L \left(\text{PT}_{I_n \rightarrow L} \left(\lfloor K \rfloor + \frac{1}{\theta} (\mathbb{K}^\theta - I_n) \right) \right) \\ &= \text{Exp}_L \left(\lfloor K \rfloor + \frac{1}{\theta} \mathbb{L}^{1-\theta} (\mathbb{K}^\theta - I_n) \right) \\ &= \lfloor L \rfloor + \lfloor K \rfloor + \mathbb{L} \left(I_n + \theta \mathbb{L}^{-1} \left(\frac{1}{\theta} \mathbb{L}^{1-\theta} (\mathbb{K}^\theta - I_n) \right) \right)^{\frac{1}{\theta}} \\ &= \lfloor L \rfloor + \lfloor K \rfloor + (\mathbb{L}^\theta + \mathbb{K}^\theta - I_n)^{\frac{1}{\theta}}. \end{aligned} \quad (\text{B.555})$$

In the third row of Eq. (B.555), the well-definedness of the exponential map requires

$$\mathbb{L} + \theta \left[\frac{1}{\theta} \mathbb{L}^{1-\theta} (\mathbb{K}^\theta - I_n) \right] \in \text{Diag}^+(n) \Leftrightarrow \mathbb{L}^\theta + \mathbb{K}^\theta - I_n \in \text{Diag}^+(n). \quad (\text{B.556})$$

For the gyromultiplication, substituting Eqs. (B.552) and (B.554) into Eq. (2.78) gives

$$\begin{aligned} t \odot L &= \text{Exp}_{I_n} (t \text{Log}_{I_n}(L)) \\ &= \text{Exp}_{I_n} \left(t \lfloor L \rfloor + \frac{t}{\theta} (\mathbb{L}^\theta - I_n) \right) \\ &= t \lfloor L \rfloor + (I_n + t (\mathbb{L}^\theta - I_n))^{\frac{1}{\theta}} \\ &= t \lfloor L \rfloor + (t \mathbb{L}^\theta + (1-t) I_n)^{\frac{1}{\theta}}. \end{aligned} \quad (\text{B.557})$$

In the second row of Eq. (B.557), the exponential map requires

$$I_n + \theta \left(\frac{t}{\theta} (\mathbb{L}^\theta - I_n) \right) \in \text{Diag}^+(n) \Leftrightarrow t \mathbb{L}^\theta + (1-t) I_n \in \text{Diag}^+(n). \quad (\text{B.558})$$

□

B.10.5 Proof of Thm. 174

Proof. In this proof, we assume $L, K, J \in \mathcal{L}_{++}^n$ and $s, t \in \mathbb{R}$, with all gyro operations satisfying the conditions in Thm. 173. By Thm. 205, we only need to prove the case of θ -DPM. For simplicity, we omit the superscript $\mathcal{C}$.

Axiom (G1). Eq. (2.77) implies that the identity element is the identity matrix.

Axiom (G2). We define the inverse element of L as

$$\ominus L = -1 \odot L = -[L] + (2I_n - \mathbb{L}^\theta)^{\frac{1}{\theta}}. \quad (\text{B.559})$$

Simple computations show that $\ominus L \oplus L = I_n$.

Axiom (G3). Gyroaddition in Thm. 173 indicates that

$$L \oplus (K \oplus J) = (L \oplus K) \oplus J = [L] + [K] + [J] + (\mathbb{L}^\theta + \mathbb{K}^\theta + \mathbb{J}^\theta - 2I_n)^{\frac{1}{\theta}}. \quad (\text{B.560})$$

Therefore the gyroautomorphism is the identity map, *i.e.*, $\text{gyr}[L, K] = \text{id}$.

Axiom (G4). This is a direct corollary of (G3).

Gyrocommutative law. Gyroaddition in Thm. 173 indicates that

$$L \oplus K = K \oplus L. \quad (\text{B.561})$$

Axiom (V1). This follows from gyromultiplication in Thm. 173 and Eq. (B.559).

Axiom (V2).

$$\begin{aligned} (s+t) \odot L &= (s+t)[L] + ((s+t)\mathbb{L}^\theta + (1-(s+t))I_n)^{\frac{1}{\theta}} \\ &= s[L] + t[L] + (s\mathbb{L}^\theta + (1-s)I_n + t\mathbb{L}^\theta + (1-t)I_n - I_n)^{\frac{1}{\theta}} \\ &= (s \odot L) \oplus (t \odot L). \end{aligned} \quad (\text{B.562})$$

Axiom (V3).

$$\begin{aligned} (st) \odot L &= (st)[L] + (st\mathbb{L}^\theta + (1-st)I_n)^{\frac{1}{\theta}} \\ &= (st)[L] + (s(t\mathbb{L}^\theta + (1-t)I_n) + (1-s)I_n)^{\frac{1}{\theta}} \\ &= s \odot \left[t[L] + (t\mathbb{L}^\theta + (1-t)I_n)^{\frac{1}{\theta}} \right] \\ &= s \odot (t \odot L). \end{aligned} \quad (\text{B.563})$$

Axioms (V4) and (V5). These two axioms can be directly obtained, as gyroautomorphisms are all identity maps. $\square$

B.10.6 Proof of Thm. 177

Proof. According to Nguyen and Yang [159, Thm. 2.4], gyrovector operations are preserved under Riemannian isometries. Moreover, the Cholesky decomposition is a Riemannian isometry:

$$\text{Chol} : \{\mathcal{S}_{++}^n, g^{\mathcal{S}}\} \rightarrow \{\mathcal{L}_{++}^n, g^{\mathcal{C}}\}. \quad (\text{B.564})$$

□

B.10.7 Proof of Thm. 179

Proof. Substituting the associated operators in Sec. 6.3.4 into Thm. 195 yields the results. □