Molecular LLM Agents: From Architectural Design to Scientific Autonomy

Jiatong Li[1]
The Hong Kong Polytechnic University
Hong Kong, China

Wengyu Zhang[2]
The Hong Kong Polytechnic University
Hong Kong, China

Weida Wang[3]
Shanghai AI Lab
Shanghai, China

Yuxuan Ren[4]
National University of Singapore
Singapore

Wei Liu[5]
Shanghai Jiao Tong University
Shanghai, China

Chenyang Mao[6]
The Hong Kong Polytechnic University
Hong Kong, China

Yuqiang Li[7]
Shanghai AI Lab
Shanghai, China

Yatao Bian[8]
National University of Singapore
Singapore

Changmeng Zheng[9*]
The Hong Kong Polytechnic University
Hong Kong, China

Xiaoyong Wei[10*]
The Hong Kong Polytechnic University
Hong Kong, China

Qing Li[11]
The Hong Kong Polytechnic University
Hong Kong, China

Abstract

Molecular science represents an important frontier for LLM-based agents. Unlike general agents that mainly operate over natural language, code, or web environments, *molecular LLM agents* must perceive, reason about, and act upon chemical objects across symbolic strings, molecular graphs, 3D conformations, spectra, simulations, and wet-lab measurements. Their capabilities depend on chemically faithful molecular perception, an LLM-centered agent framework, domain-specific tool grounding, and computational or experimental feedback, in addition to planning and tool use. This work develops a conceptual framework for molecular LLM agents from two complementary perspectives. **First**, we introduce an architectural view of molecular-agent design, covering molecular representation and perception, the agent framework, domain-specific toolboxes, and learning and optimization. **Second**, we propose a scientific autonomy ladder inspired by staged autonomy in engineering systems, categorizing agents into four levels: **L1** assistive or fixed workflows, **L2** adaptive computational agents, **L3** feedback-aware physical experiment agents, and **L4** scientific-agenda agents. Together, these two perspectives establish a comprehensive framework for comparing existing molecular LLM agents, identifying missing capabilities and deployment risks, and guiding the design, evaluation, and deployment of future agents in molecular discovery workflows.

CCS Concepts

• **Applied computing** → **Bioinformatics**.

Keywords

Molecular LLM agents, Molecular discovery, Scientific agents, Autonomous laboratories

1 Introduction

Molecular discovery asks how to turn a desired function into a real molecular entity [65, 66]. The target may involve biological activity, selectivity, toxicity, stability, or other physicochemical and functional properties [46, 154]. A useful molecule must also be chemically valid, compatible with multiple objectives, synthetically accessible, safe for its intended context, and supported by computational or experimental evidence [11, 110]. The search space is vast, evaluation is costly and uneven, and progress depends on coordinating the appropriate representation, model, tool, and validation signal.

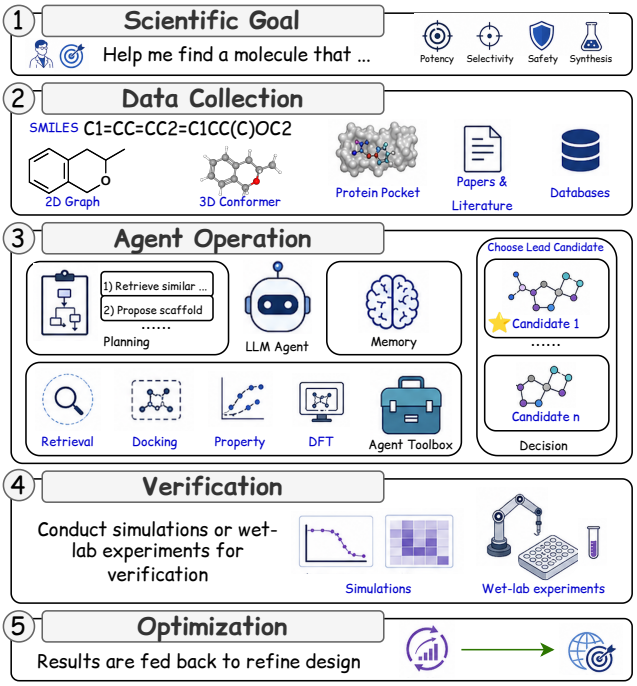

Figure 1: End-to-end workflow of a molecular LLM agent. Starting from a scientific goal and task-specific constraints, the agent gathers multi-modal molecular evidence, plans and executes chemistry-tool operations, selects candidates, and verifies them through simulation or wet-lab experiments. Verification results are then fed back to refine the molecular design and the agent's subsequent actions.

Author emails (numbers match the author list):
[1] jiatong.li@connect.polyu.hk; [2] wengyu.zhang@connect.polyu.hk;
[3] wangweida@pjlab.org.cn; [4] yuxuan.ren@nus.edu.sg;
[5] captain.130@sjtu.edu.cn; [6] 25047313g@connect.polyu.hk;
[7] liyuqiang@pjlab.org.cn; [8] ybian@nus.edu.sg;
[9*] changmeng.zheng@polyu.edu.hk; [10*] x1wei@polyu.edu.hk;
[11] csqli@comp.polyu.edu.hk.
[*] *Corresponding authors: Changmeng Zheng and Xiaoyong Wei.*

Modern molecular AI has strengthened many components of this process. Graph-based models have improved structure-property prediction [158], while graph generative models support molecular generation and optimization in graph space [51]. Chemical language models such as ChemBERTa [20] and MoLFormer [114] learn molecular representations from SMILES [151]. MolT5 [26] and MolReGPT [67] further connect molecular structures with natural language for captioning, generation, and in-context learning. Together with reaction prediction, synthesis planning, and molecular optimization, these methods provide a rich stack of capabilities. Most nevertheless remain specialized models or fixed mappings, such as molecule to property, text to molecule, or candidate to score.

The practical bottleneck is therefore shifting from solving an individual subtask to orchestrating subtasks into a coherent discovery workflow [70]. A chemist must translate a natural-language objective into molecular objects, choose among strings, graphs, conformers, and spectra, retrieve prior evidence, invoke scientific tools, reject invalid candidates, interpret noisy outputs, and decide what should happen next. Conventional molecular AI leaves most of this orchestration to the user. A model may predict, generate, or rank, while the user still decides which evidence to trust, how to repair failures, and when to continue, stop, or proceed toward synthesis and measurement.

Large language model (LLM) agents offer a route from isolated models to action-oriented molecular systems. The LLM acts as a controller that interprets a goal, decomposes it into actions, invokes tools, observes feedback, and revises the plan. ReAct [162] made this control pattern explicit by interleaving reasoning with actions, while Toolformer [118] demonstrated language-model tool invocation. In molecular discovery, an LLM-centered system may strengthen one subtask through reasoning [71, 149], retrieval or verification [68], or coordinate a longer trajectory involving structure lookup, property calculation, docking, quantum chemistry, molecular dynamics, retrosynthesis, and laboratory interaction. The key change is that the model participates in making decisions, selecting actions and interpreting observations rather than stopping after a single prediction or generation step.

The molecular setting makes this control problem distinct from general web or software agents [99]. The agent must preserve chemical identity, connectivity, stereochemistry, geometry, units, and experimental conditions while moving among representations and tools. Its observations range from inexpensive heuristic scores to simulations, spectra, assays, and hardware logs, each with different uncertainty and cost. Errors can therefore propagate from a textual decision into an invalid calculation, an expensive simulation, or a physical experiment. Fluent reasoning alone is not sufficient; actions and feedback must remain chemically grounded and auditable.

We use *molecular LLM agent* to denote an LLM-centered decision system that operates on molecular states, reasons over scientific goals, selects actions through chemistry tools or environments, and uses computational or experimental observations to produce, validate, or refine molecular outcomes. Figure 1 summarizes this feedback-driven workflow from goal specification and data collection to agent operation, verification, and design revision. ChemCrow [91] coordinates chemistry tools for structure lookup, property calculation, reaction prediction, and synthesis planning. Coscientist [10] extends the pattern toward laboratory-facing execution and feedback. These systems illustrate the move from task-specific molecular models to controllers over perception, action, tools, and evidence.

Despite this progress, the field lacks a shared design framework, autonomy roadmap, and governance boundary. Existing systems differ in molecular representation, tool interfaces, memory, feedback, optimization, and human oversight, yet are often discussed under the same agent label. This ambiguity leaves three questions unresolved. Which components should a molecular agent contain, and how should they interact? How could fixed workflows, computational loops, physical experimentation, and open-ended scientific agency be distinguished? And how could increasingly consequential actions be evaluated and governed? Without a common framework, it is difficult to compare systems, identify missing capabilities, or assess deployment risks.

We address these questions through two complementary perspectives. **The architectural view** decomposes molecular agents into molecular representation and perception, an LLM-centered agent framework, domain-specific toolboxes, and learning and optimization. It treats evaluation, safety, and trustworthy deployment as cross-cutting requirements rather than properties of the LLM alone. Section 3 separately consolidates evaluation settings for component capabilities, executable workflows, research tasks, and closed-loop discovery. **The autonomy view** classifies systems by the outermost feedback loop they demonstrably close: L1 assistive or fixed workflows, L2 adaptive computational agents, L3 feedback-aware physical workflows, and L4 scientific-agenda agents. Section 4 provides the operational definitions and boundary cases.

Our contributions are as follows:

- We conceptualize molecular LLM agents as action-oriented systems that connect molecular state, agent control, scientific tools, and feedback.
- We develop a dual-perspective framework that links architectural design to an evidence-based L1 to L4 scientific-autonomy ladder.
- We use the framework to compare existing systems, identify capability and evaluation gaps, and characterize deployment and governance risks.

2 Architecture Design of Molecular Agents

We organize a molecular LLM agent into four interacting components, as shown in Figure 2. The *perception layer* represents molecular objects and routes them among strings, graphs, geometries, images, and structured records. The *agent framework* reasons over scientific goals, plans actions, maintains memory, reflects on feedback, and may coordinate specialized agents. The *toolbox* grounds these decisions in databases, cheminformatics, simulation, synthesis, and experimental interfaces. Finally, *reflection, learning, and optimization* convert computational or physical feedback into revisions of molecular candidates, plans, or reusable policies. These components form a closed chain from molecular state to decision, action, observation, and optimization.

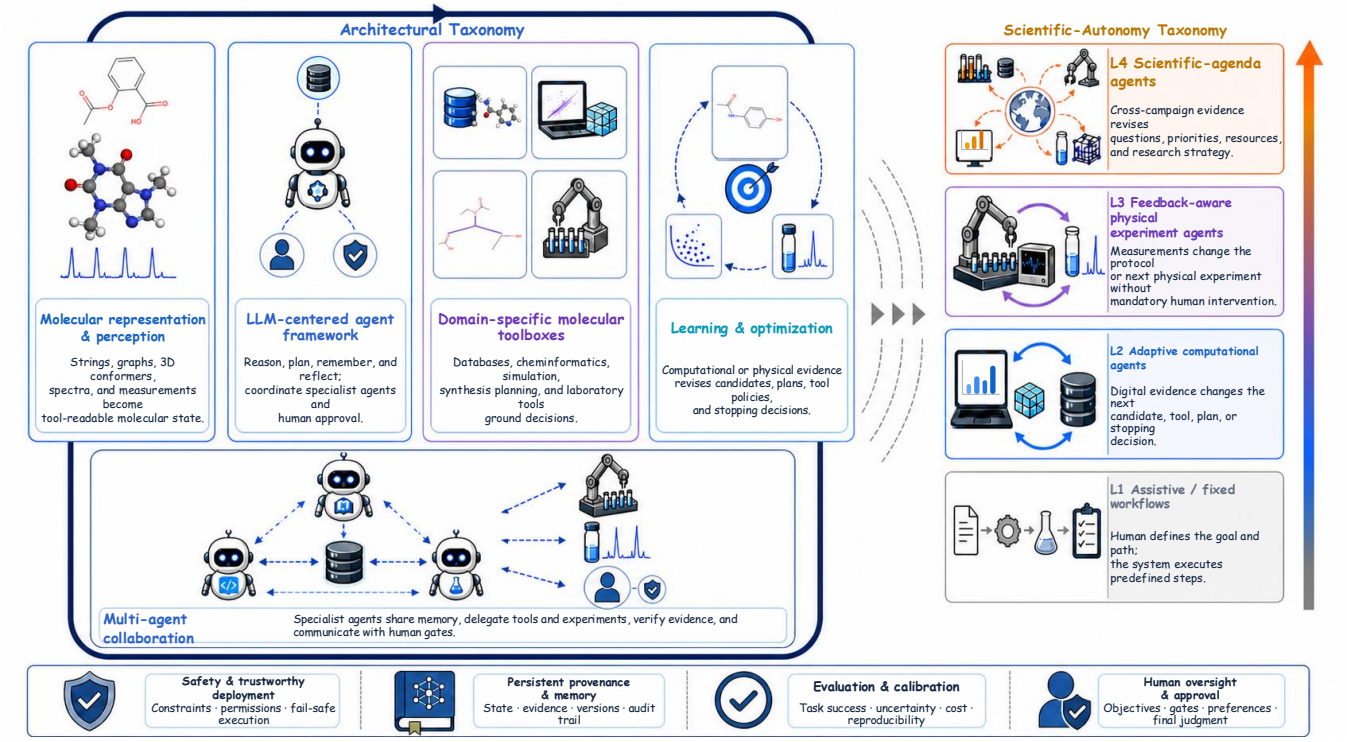

Figure 2: Two complementary taxonomies for molecular LLM agents. The architectural taxonomy (left) organizes agent systems around molecular representation and perception, an LLM-centered agent framework, domain-specific molecular toolboxes, and learning and optimization, together with multi-agent collaboration and cross-cutting deployment requirements. The scientific-autonomy taxonomy (right) classifies systems by the outermost feedback loop they can reliably close without mandatory human intervention, ranging from L1 assistive or fixed workflows to L4 scientific-agenda agents.

2.1 Molecular Representation and Perception

Perception is the interface through which a molecular agent turns a chemical object into a state that an LLM-centered controller can read, edit, verify, and pass to tools. This interface is more constrained than ordinary text perception: a molecule has atom identities, bond orders, aromaticity, charge, stereochemistry, conformers, electronic effects, and task-dependent physical context. Consequently, the agent does not perceive "the molecule itself", but a representation of it. The choice of representation influences which chemical facts are explicit, which facts must be inferred, and which actions are comparatively easy or brittle; these effects also depend on the model, task, and validation tools surrounding the representation.

2.1.1 Representation Substrates: Strings, Graphs, Geometry, Images, and Structured Text. The most common entry point is a 1D chemical line notation. SMILES serializes a molecular graph into an ASCII string and remains attractive because it is compact, parser-friendly, and directly compatible with sequence models [151]. InChI offers another standardized identifier oriented toward chemical databases and interoperability [42]. For LLM agents, these strings are convenient action interfaces: a controller can generate a candidate SMILES, call a toolkit to parse it, compute descriptors, search a database, or send it to a retrosynthesis service. Their weakness is that the graph is implicit in a traversal. Branches, ring closures, aromaticity, and stereochemical marks must be reconstructed from the sequence, and different valid strings can describe the same molecule. This is acceptable for fast screening and tool calls, but fragile for tasks that require exact topology editing [72] or long-range structural reasoning.

SELFIES changes the failure mode by defining a robust string representation in which every SELFIES string maps to a chemically valid molecule under the representation's constraints [58]. This makes SELFIES especially useful when an agent repeatedly samples, mutates, or optimizes molecular strings, because local generation errors are less likely to collapse into unparsable outputs. The guarantee, however, is a validity guarantee rather than a guarantee of stability, synthesizability, or task quality. Fragment-level variants move the perceptual unit closer to how chemists design molecules. Group SELFIES extends SELFIES with group tokens for common functional groups or substructures [18], while SAFE represents a molecule as an unordered sequence of connected fragment blocks that remains compatible with SMILES parsers [100]. MolLingo and mCLM continue this direction by using molecule-native or synthesis-friendly building blocks for LLM-powered agents and chemical language models [25, 98]. These representations are less character-centric and more action-centric: the agent can operate

on scaffolds, linkers, substituents, motifs, or functional modules instead of only on individual symbols.

Two-dimensional molecular graphs expose the topology directly. Atoms become nodes, bonds become edges, and node or edge features encode element type, valence, charge, aromaticity, bond order, and stereochemistry. This makes graph encoders natural for property prediction and structure-aware generation; directed message passing is a representative example of this family [158]. The difficulty is that a graph is not natively an autoregressive text sequence. An LLM agent therefore needs either a graph encoder whose continuous output is projected into the language model, or a discrete graph-to-text interface that turns nodes and bonds into tokens. Recent structured molecular languages pursue the latter route. MolJSON encodes atoms and bonds in a JSON schema designed for LLM reasoning [117], and MoleCode uses explicit node-edge-subgraph primitives so that connectivity and stereochemical information are visible in the context window [157]. The broader lesson from these works is that structure-sensitive reasoning improves when the relational structure is not hidden inside a linearized string.

Three-dimensional representations add geometry. Coordinates, distance matrices, conformer ensembles, protein pockets, docking poses, or crystal structures are necessary when the relevant property depends on spatial arrangement rather than only connectivity. SchNet showed how neural models can learn from interatomic distances for quantum interactions [119], and GraphMVP aligns 2D graph representations with 3D geometry to transfer conformational information into molecular graph encoders [83]. For agents, 3D perception is expensive but often decisive: docking, molecular dynamics, transition-state reasoning, binding-pose inspection, and materials simulation require coordinates and physical units that a line notation does not provide. The practical problem is that a molecule may have many low-energy conformers, and downstream decisions can be sensitive to how those conformers are generated, ranked, and passed between tools.

Images provide a fourth perceptual channel [73]. A rendered skeletal formula is close to what chemists see in papers, patents, and lab notebooks. MolScribe treats molecular structure recognition as image-to-graph generation [112], and MolSight studies progressive visual pretraining over molecular diagrams for property prediction [7]. Image perception is valuable when the agent must read literature or multimodal records, but it also introduces optical recognition errors: atom labels, wedge bonds, charges, and crowded ring systems must be converted back into a valid graph before most tools can use them.

Experimental observations form a further perceptual substrate once an agent interacts with instruments or laboratory records. Spectra, chromatograms, assay tables, images, time series, protocol events, and robot logs describe a molecular system under particular conditions rather than molecular identity alone. They therefore require the agent to preserve units, acquisition settings, sample identifiers, uncertainty, and provenance together with the measured values. Coscientist interprets UV–Vis measurements inside a physical task [10]; LLM-RDF uses reaction yields and spectral analysis in an end-to-end synthesis-development workflow [116]; and ORGANA combines visual feedback with experiment execution and reporting [22]. These systems illustrate why experimental records should be represented as state-bearing observations, not flattened into unqualified text.

2.1.2 Tokenization: The Agent's Molecular Action Interface. Once a molecule is written as text or structured text, the LLM still sees only token IDs. For LLM-based molecular agents, tokenization is not a neutral preprocessing step: it defines the granularity at which the agent can act on molecules, the way errors propagate through tool chains, and whether the agent can later read and reflect on its own molecular edits. The question is distinct from tokenization for specialized chemical language models such as ChemBERTa [20], MoLFormer [114], or SELFormer [169], which train dedicated encoders on domain-specific corpora and can choose tokenizers optimized for their own architecture. An LLM agent, by contrast, typically operates through a general-purpose language model whose tokenizer is fixed at pretraining. Molecular strings must pass through this given tokenizer and remain actionable on the other side. The agent-level consequences of this constraint vary across autonomy levels.

Character-level molecular tokens and the L1 baseline. A generic subword tokenizer (BPE, WordPiece, Unigram) trained on natural-language corpora can split multi-character elements such as `Cl` and `Br`, scatter bracketed atoms and stereochemical marks across several tokens, and merge chemical punctuation with atoms in arbitrary ways [135]. At L1, where a human or fixed workflow closes each consequential transition, this can be tolerable: the agent generates a SMILES candidate, and a human review or deterministic validation catches parsing errors before they reach a downstream tool. Atom-wise tokenizers that preserve atom boundaries and bracketed atoms improve this baseline, and Smirk decomposes bracket atoms into glyph-level tokens to retain open-vocabulary coverage for organometallics, coordination complexes, and other chemistry that general tokenizers replace with undifferentiated unknown tokens [147]. Even at L1, coverage matters because agents may query patents, catalysts, materials, metal complexes, polymers, salts, and tool-produced intermediates whose chemistry falls outside the clean organic subset seen during pretraining.

Fragment-level tokens and L2 iteration efficiency. When the agent moves to L2 and autonomously iterates through cycles of proposal, evaluation, and revision, tokenization begins to shape the optimization dynamics. A character-level action edits one atom symbol; replacing a substituent on a scaffold may require five to eight sequential edits. Fragment-level tokenizers, including Group SELFIES [18], SAFE [100], and mCLM [25], elevate the action unit to a chemically coherent motif such as a functional group, scaffold, or linker. The same edit can then be expressed in a single step. This can shorten the action sequence needed to express a scaffold-level edit and may simplify search or credit assignment when the model and objective are aligned with the chosen fragments. The realized gain in planning or computational efficiency is system-dependent, however, and should not be attributed to token granularity alone. Fragment-level tokenization also changes the error profile. At the character level, a mismatched ring-closure digit or unbalanced bracket can cause an RDKit [8] parsing failure. Descriptor computation and docking preparation then cannot proceed, so the workflow halts. The agent must detect the failure, diagnose its source, and retry, consuming

Table 1: Common molecular and experimental perceptual substrates for LLM-based molecular agents.

Substrate	Typical form	What it makes explicit	Agent-level use and main risk
Line notation	SMILES, InChI	Compact identity and parser-compatible connectivity	Fast generation, search, and tool calls; topology and stereochemistry must be recovered from a brittle sequence.
Robust or fragment string	SELFIES, Group SELFIES, SAFE, molecule-native fragments	Validity constraints or chemically meaningful blocks	Useful for repeated editing and optimization; validity does not ensure synthesizability or property quality.
Graph or structured text	Molecular graph, MolJSON, MoleCode	Atoms, bonds, local topology, explicit node and edge identities	Better for exact edits and graph reasoning; requires graph encoders or verbose structured contexts.
3D geometry	Conformers, coordinates, pockets, docking poses	Distances, angles, chirality, spatial contacts, physical state	Necessary for binding, quantum, and simulation tasks; expensive and conformer-sensitive.
Image	Skeletal diagram or scanned chemical figure	Human-readable structural drawing and visual stereochemistry cues	Connects agents to papers and notebooks; must be recognized into graph or string form before most tools can act.
Experimental record	Spectra, chromatograms, assays, time series, protocol events, robot logs	Measured response, conditions, units, acquisition context, and execution state	Grounds interpretation and laboratory control; sample mismatch, lost metadata, or overconfident peak assignment can corrupt later decisions.

planning budget. SELFIES's validity guarantee [58] and fragment-level constraints reduce the space of syntactically malformed outputs. They can therefore limit one source of error propagation at the perception-to-tool interface, while leaving stability, synthesis feasibility, and task quality unresolved. This benefit may be missed by language-modeling evaluation but can matter for L2 agents that generate and evaluate many candidates in a campaign. Tool augmentation can otherwise degrade chemistry problem solving when inputs are malformed [78, 166].

Structure-discretized tokens and the transparency-efficiency tradeoff. A newer family of tokenizers discretizes molecular structure rather than strings. UniMoT uses a molecule encoder, a causal Q-Former, and vector quantization to map molecular graph features into discrete tokens that can be added to an LLM vocabulary [39]. AtomDisc and VQ-Atom similarly use graph or geometry context to quantize atom-level chemical neighborhoods [56, 175]. For agents, this can reduce the interface gap between continuous molecular encoders and symbolic action traces and compress complex substructural edits into single token predictions. The cost is reflective opacity. Agents reflect on their action trajectories by reviewing previous tool calls, diagnosing failures, and deciding what to retry [75, 133]. This reflection operates in the LLM's text space: character-level and fragment-level tokens are human-readable, so the agent and a human auditor can inspect which molecular edit was attempted and what went wrong. Structure-discretized tokens are opaque. An agent may record "I predicted code 437 → code 892" without being able to express the corresponding structural change unless it invokes an external decoder. MolLingo [98] proposes molecule-native representations that retain structural meaning, while LatentChem [164] replaces verbose textual chain-of-thought with latent thinking; both illustrate the design tension. At L2, where the agent must self-correct during iterative optimization, reflective opacity can be a significant bottleneck.

2.1.3 Perception as a Routing and Interaction Layer. Current molecular agents rarely rely on a single representation throughout a task. They route among representations according to the action being taken: generating a SMILES candidate, parsing it into a graph for editing, converting to 3D for docking, converting back to SMILES for a database query, and rendering an image for human review. ChemCrow couples an LLM controller to chemistry tools so that natural-language requests can be translated into such operations [91], and ChemAgent adds memory and tool-use policies so that the perceived state includes prior tool outputs, evidence, and task history [133]. Routing begins with entity resolution. A user-supplied name, CAS number, or drawn structure is mapped to a canonical molecular representation through name-to-structure conversion with resources such as OPSIN [89] or PubChem [55], followed by protonation-state assignment and tautomer selection. Routing continues through every later representation transition. At L1, a human or fixed workflow directs each transition and catches errors before they propagate; the routing layer is effectively a bidirectional format translator. As autonomy rises, routing must become an autonomous state-management system with two capabilities that are largely unaddressed in existing architectures: conversion fidelity and bidirectional tool-facing perception.

Conversion validation. Each representation transition can lose information: stereochemistry may not survive a SMILES-to-graph-to-SMILES round trip; a conformer generator may fail silently; a structured-text encoding may truncate features that exceed the context-window budget. Without validation checkpoints at each conversion boundary, the agent may silently operate on a different molecule from the one it started with. MolJSON [117] and MoleCode [157] make relational structure visible in the context window, which supports conversion-aware perception. Most agent architectures still lack systematic checks that atom count, bond count, stereochemistry, and charge survive each transition. At L2, where the agent controls the routing, these checkpoints become mandatory infrastructure; a single undetected loss in cycle 3 of a 10-cycle optimization corrupts all subsequent cycles. At L3, conversion validation extends to physical signals: the perception layer must verify both structural preservation and correct parsing, compound assignment, and condition annotation for each experimental readout.

Output perception and feedback perception. Before invoking a tool, the agent must transform its internal state into the tool's expected input format: a 3D SDF file for docking, canonical SMILES for a database query, charge and spin multiplicity for quantum chemistry [182], or a valid graph with explicit hydrogens for retrosynthesis. Many agent failures originate at this output-perception boundary: the LLM reasons correctly about which tool to call but constructs an invalid input. ChemHAS [78] and TRACE [75] address this through self-correction at the tool interface, but recognizing these as perception failures rather than reasoning failures enables more targeted evaluation. In the reverse direction, a tool output

such as a docking score, ADMET prediction, DFT energy, or spectral match must return to the agent's reasoning space with context. The agent must determine whether the output corresponds to the intended molecule, quantify its uncertainty, and identify conflicts with other signals. Tooling-or-Not-Tooling [166] shows that tool augmentation can help or hurt depending on context. The perception layer should therefore assess the reliability of each feedback signal before incorporating it into the agent state. At L3, feedback perception faces a distinct challenge: experimental signals are inherently ambiguous, and the perception layer must preserve this ambiguity rather than collapsing to a single interpretation, since premature disambiguation can produce false experimental conclusions. At L4, feedback from different campaigns, tools, and scoring functions must be compared on a common perceptual basis, which requires unified state representations and provenance tracking across projects.

Thus, molecular perception should route among multiple representations, remain aware of tokenizer effects, validate conversions explicitly, and support bidirectional interaction with tools. The downstream controller can plan reliably only when its perceptual state preserves the chemical constraints needed by the next action. This requirement scales from human-checked format translation at L1 to cross-campaign state unification at L4.

2.2 Agent Framework

The agent framework is the LLM-centered controller between molecular representation and scientific action. It selects tools or subagents, interprets their outputs, and decides whether a trajectory should continue, revise, stop, or request approval. ChemCrow [91] and CACTUS [93] illustrate chemistry-tool control; MDCrow [13] and LLaMP [19] extend it to longer computational workflows; and Coscientist [10], LLM-RDF [116], and Tippy [27, 28] move toward physical or laboratory-facing decisions.

2.2.1 Overview: The Molecular Controller. A molecular controller is judged by the scientific objects and actions it can produce, not by fluent text alone. Its outputs must remain chemically valid, its decisions must be grounded in noisy or costly evidence, and its internal state must preserve the molecular representation needed by downstream tools. Most systems implement an observe, plan, act, and revise loop through five faculties: reasoning, planning, memory, reflection and self-correction, and multi-agent collaboration.

Figure 3 cross-tabulates papers by these faculties and by the highest demonstrated autonomy level under Section 4. Each paper receives one global level that is reused across faculties: the heatmap counts it once per relevant faculty, whereas the yearly bars count it once overall.

2.2.2 Reasoning. Reasoning turns a molecular goal into hypotheses, intermediate conclusions, and candidate actions. Early chemistry agents often express this reasoning as natural-language chain-of-thought over names, SMILES strings, reaction descriptions, or retrieved text. Such reasoning is interpretable, but it is fragile: a small mistake in valence, stereochemistry, units, reaction feasibility, or tool-input syntax can invalidate a fluent explanation. Chemistry-oriented LLMs therefore strengthen the reasoning substrate through domain training, memory, and molecular language modeling, as illustrated by ChemAgent [133], ether0 [95], mCLM [25], and comprehensive molecular design language models [168].

One direction replaces linear reasoning with search-augmented reasoning. Molecular design is combinatorial, and many tasks require comparing multiple hypotheses rather than committing to one generated chain. Monte Carlo Thought Search [127] explores catalyst-design reasoning paths with tree search, while ChemReasoner [128] searches an LLM's chemistry knowledge space and grounds the search with quantum chemical rewards. CheMatAgent [152] learns chemistry and materials tool-use policies through tree search based training, DrugMCTS [161] combines retrieval, multi-agent roles, and Monte Carlo Tree Search for drug repurposing, and Agents-on-a-Tree [173] coordinates pathwise molecular optimization. A molecular reasoning branch can therefore be evaluated by docking, property prediction, adsorption energy, reaction barriers, quantum calculations, or other physical feedback instead of linguistic plausibility alone.

A second direction makes reasoning structured and executable. Instead of producing only plain text, the agent emits tool calls, action sequences, code, protocol steps, or parameterized workflows that can be parsed, executed, checked, or replayed. ChemActor [177] converts unstructured synthesis procedures into machine-executable chemical action sequences, MT-Mol [54] decomposes molecular optimization into tool-based reasoning, and DrugPilot [74] uses parameterized reasoning over multimodal drug-discovery information. Coscientist [10], El Agente [182], DrugAgent [82], and Chemist-X [16] expose reasoning through executable code, workflow traces, or computer-aided design interfaces.

A third direction asks whether the agent should reason in text at all. Text is inspectable, but it is lossy for graphs, conformers, stereochemical relations, and 3D interactions. MolLingo proposes molecule-native representations for LLM scientific agents [98], and LatentChem replaces verbose textual chain-of-thought with latent thinking and a dynamic perception loop [164]. This creates a design trade-off: text supports human inspection, whereas molecule-native or latent states may preserve structural information that is hard to express in prose. The boundary between reasoning and tool use is also adaptive rather than fixed, since tool augmentation can help some chemistry tasks and hurt others [166]. Reasoning becomes autonomy-relevant only when its evidence changes a later scientific action. ChemNavigator [108], for example, extracts design rules within a bounded computational campaign and is therefore an L2 boundary case rather than evidence of cross-campaign L4 discovery.

2.2.3 Planning. Planning turns a reasoning outcome into an executable trajectory. For molecular agents, a plan must specify how information moves across representations, tools, constraints, and feedback signals. A goal may require literature retrieval, candidate generation, filtering, docking, retrosynthesis, simulation, protocol generation, or laboratory execution. Thus, planning lengthens the agent's horizon from one-shot tool use to in-silico campaigns.

A common pattern is task decomposition and workflow orchestration. M^{4}olGen [76] studies multi-stage generation under precise multi-property constraints, while Prompt-to-Pill [144], PharmAgents [30], MADD [125], and FROGENT [105] organize broader

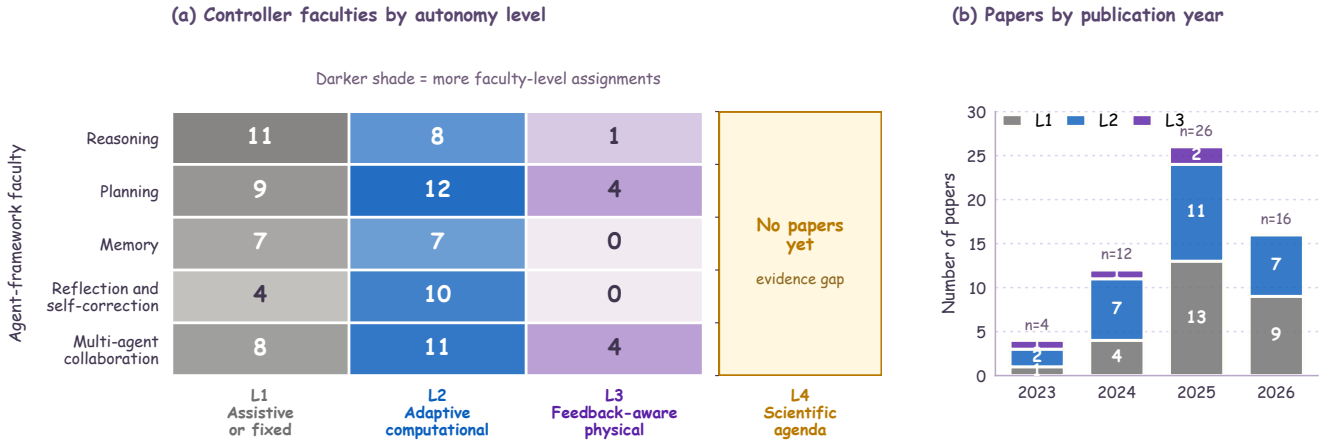

Figure 3: Two views of the 58 surveyed papers by highest demonstrated autonomy level. The faculty heatmap counts 96 paper–faculty assignments, while the yearly stacked bars count each paper once; hatching marks the L4 evidence gap.

drug-discovery pipelines into stages or agents. These systems show that generation, scoring, synthesis analysis, and reporting work closely together: each stage changes the state the planner must pass to the next module.

Another pattern is interleaved reason-act planning. Rather than drafting a complete plan once, the agent alternates between local reasoning, tool invocation, observation, and revision. ChemCrow [91] and CACTUS [93] exemplify this mode for chemistry tasks, LLaMP [19] extends it to materials retrieval and simulations, and MDCrow [13] applies it to molecular-dynamics workflows. Longer tasks motivate hierarchy: El Agente [182], MASTER [115], robotic ChemAgents [126], and Tippy [27, 28] separate high-level scientific intent from mid-level workflow control and low-level execution.

Planning also includes tool selection and action-policy design. In molecular agents, choosing RDKit, docking, retrosynthesis, DFT, molecular dynamics, a database, or a verifier is already a scientific decision. ChemHTS [77] studies hierarchical tool stacking, and ChemHAS [78] improves chemistry-tool performance through agent stacking. TRACE [75] frames lead optimization as resource-aware planning, CheMatAgent [152] learns tool-use policies, and Mozi [14] represents governed autonomy through state-aware skill graphs. The planner should therefore be treated as a policy over admissible scientific actions, not as a free-form text generator that happens to call tools.

The planning loop closes at different levels. At L2, agents propose candidates or workflows, evaluate them with predictors or simulators, and refine the next action, as in dZiner [4], ChatMOF [53], and MDCrow [13]. At L3, Coscientist [10] and ORGANA [22] demonstrate planning and physical execution with feedback-aware recovery, perception, or analysis. LLM-RDF [116] and robotic ChemAgents [126] demonstrate the stronger iterative form of L3 because measured reaction yields or catalyst performance determine a subsequent physical experiment. AutoLabs [106] remains L2 under our evidence rule: it evaluates multi-agent protocol generation and self-correction, but reports a critical human verification step before instrument execution. Tippy [27, 28] also remains L2 because its reported evaluation does not establish autonomous physical execution. At L4, planning would become research-program design, where agents select problems, allocate resources, compare campaigns, and extract reusable rules.

2.2.4 Memory. Memory gives the controller persistence beyond the current prompt. Molecular discovery may involve repeated molecule edits, failed tool calls, changing constraints, simulation parameters, synthesis attempts, assay evidence, and human decisions. A context window can hold part of this state, but it does not provide durable, searchable, provenance-aware scientific memory. ChemAgent [133] uses self-updating memories to improve chemical reasoning, while modular drug-discovery agents [102] and TRACE [75] show why long-horizon tasks require preserving intermediate state and action history.

Molecular memory is structured rather than conversational. It may store molecules, conformers, pockets, score vectors, docking poses, reaction conditions, failed candidates, tool parameters, literature snippets, assay records, protocol versions, and design rationales. ChemAgent [133] is a representative typed-memory architecture with planning, execution, and knowledge memories; DrugPilot [74] maintains a parameterized memory pool for multimodal drug-discovery information; and El Agente [182] uses memory inside a hierarchical quantum-chemistry workflow.

Retrieval-augmented generation is another form of external memory. Chemist-X [16] retrieves literature and database evidence for reaction-condition recommendation, ChatDrug [84] combines retrieval with domain feedback for conversational drug editing, and RAG-enhanced collaborative agents [62] use retrieval for drug-discovery reasoning. Agent-based learning from literature [3] extracts structured materials data from papers, while DrugAgent [48] and DrugMCTS [161] illustrate knowledge-graph and retrieval memory for drug-target or repurposing tasks. Provenance is essential: the agent should retain a fact together with its source and any later contradictory evidence.

Experience memory connects agent design to optimization. Previous failures can prevent repeated invalid edits, bad docking setups,

weak analogs, and tool-instruction errors. TRACE [75] stores instruction and result histories as action-level experience, Augmented Memory [38] uses experience replay for sample-efficient de novo design, ExLLM [113] uses experience-enhanced optimization, and Mozi [14] stores reusable procedures in skill graphs. Memory also introduces risks: stale literature, noisy proxy scores, contaminated examples, and incorrect tool outputs can be amplified if the controller does not decide what to write, retrieve, forget, and trust. These memory functions support progressively longer and more reproducible workflows, but stored context alone does not raise autonomy unless it changes a later scientific decision.

2.2.5 Reflection and Self-Correction. Reflection converts feedback into correction. In general LLM agents, reflection often means textual self-critique. For molecular agents, this is insufficient because a fluent critique may miss an invalid structure, infeasible reaction, wrong unit, malformed docking input, unstable simulation, or unsafe protocol. A molecular controller should therefore reflect against external signals such as validity checks, property predictors, docking, retrosynthesis, spectra, assays, failed tool calls, hardware feedback, and expert review.

A major use is iterative molecular editing and optimization. ChatDrug [84] combines conversational editing with retrieval and domain feedback, AgentDrug [61] uses domain feedback to steer zero-shot molecular optimization, and Probe-Before-You-Edit [160] uses structure-based feedback before editing molecules. DrugAssist [163], GeLLM^{3}O [23], and ExLLM [113] similarly condition later proposals on observed weaknesses of earlier candidates. In these systems, reflection is not an explanation after generation, but part of the optimization dynamics.

Another pattern separates generation from verification. MT-Mol [54] uses specialized tool-based roles including verifier and reviewer functions; ChemActor [177] adds structured action sequences and multi-round review for synthesis extraction; and ChemLabs [156] uses multi-agent checking for multimodal chemistry reasoning. Debate is a stronger variant: Mol-Debate uses disagreement among agents to improve molecular structural reasoning [176], and collaborative expert LLMs expose trade-offs in multi-objective optimization [167]. However, debate is useful only when agents bring diverse evidence, tools, or objectives; otherwise it can amplify shared errors.

Reflection also applies to tool failures. Invalid SMILES strings, malformed arguments, missing database fields, incompatible files, unstable simulations, and contradictory scorers can all corrupt downstream planning. ChemHAS applies self-correction at the chemistry-tool interface [78], and TRACE reuses previous tool-instruction failures to refine future actions [75]. AutoLabs [106] extends self-checking to the generation of hardware-ready experimental procedures, but its reported pre-execution human verification gate keeps the evaluated workflow at L2 under our rubric. The autonomy contribution of reflection therefore depends on the source of the feedback and on which later decision it is allowed to revise, not on the presence of a self-critique step.

2.2.6 Multi-Agent Collaboration. Multi-agent collaboration distributes controller functions across specialized roles. This design is natural because discovery spans heterogeneous artifacts and skills: literature, structures, protein pockets, reaction schemes, simulations, spectra, protocols, and lab readouts. A single controller may struggle to maintain all contexts, whereas specialized agents can separate literature search, generation, scoring, synthesis planning, analysis, verification, and human communication. The relevant design criterion is whether the division of labor matches the molecular task, not the number of agents.

Role-specialized pipelines decompose discovery into stages. PharmAgents [30], Prompt-to-Pill [144], MADD [125], FROGENT [105], and M^{4}olGen [76] assign different agents or stages to target analysis, generation, scoring, synthesis, and reporting. At L3, agents can map directly onto laboratory roles. Coscientist [10] coordinates specialized planning, search, code, and automation modules. LLM-RDF [116] assigns agents to literature scouting, experiment design, hardware execution, spectrum analysis, separation instruction, and result interpretation, while robotic ChemAgents [126] coordinates literature, experiment design, computation, and robotic execution in an iterative materials campaign. ORGANA [22] combines task planning, visual feedback, robot control, and reporting. AutoLabs [106] and Tippy [27, 28] define useful laboratory-facing roles but remain L2 here because their reported evaluations retain a pre-execution human gate or do not establish autonomous physical execution.

Many systems use an orchestrator or supervisor. Mozi [14] maintains governed autonomy through state-aware skill graphs. MASTER [115] uses hierarchical multi-agent reasoning for functional-materials discovery. El Agente [182] organizes quantum-chemistry workflows through hierarchical control. This authority structure improves traceability, since decisions can be attributed to a supervisor, specialist, tool, or human gate, and it supports permissioning for high-risk actions.

Collaboration can also improve critique [178]. Mol-Debate [176], MT-Mol [54], ChemLabs [156], and MASTER [115] use multi-agent disagreement, verification, or peer review as quality control. Agents-on-a-Tree [173], DrugMCTS [161], multi-GPT-agent reinforcement learning [45], and collaborative expert LLMs [167] coordinate agents over shared chemical search spaces. Humans can also be first-class collaborators: collaborative structure-based drug design [31] and ORGANA [22] keep experts in the loop for objectives, approvals, preferences, and final judgment. Role specialization can support any level; the classification depends on the strongest evidence-conditioned action performed by the team as a whole.

2.2.7 Cross-Cutting Synthesis. Figure 3 contains 96 faculty-level assignments: 39 at L1, 48 at L2, and 9 at L3. Fixed tool chains, predefined role pipelines, and review-only ensembles remain L1 unless evidence changes a later scientific decision. Feedback-aware physical workflows qualify as L3, with measurement-driven experiment selection identified as iterative L3. No surveyed paper meets the L4 criterion.

The five faculties are therefore enabling mechanisms rather than autonomy levels. Their role changes with feedback source and action authority: memory progresses from retrieval to experimental state, while collaboration progresses from review ensembles to divisions of laboratory work. As authority increases, permissioning,

traceability, tool constraints, and approval gates become part of the controller itself.

2.3 Domain-Specific Molecular Toolboxes

Molecular toolboxes translate language-level plans into executable chemical operations over structures, databases, simulations, spectra, and laboratory protocols. ChemCrow [91] and CACTUS [93] illustrate chemistry-tool control, Coscientist [10] extends it toward experiments, and ToolUniverse [32] treats tool composition as reusable scientific infrastructure. The toolbox is therefore part of the agent architecture: it defines both the available actions and the observations that can enter the control loop.

Those observations differ in cost and reliability. Computational scores, assay records, spectra, and robot logs must be interpreted within their operating conditions. Tool access alone is not beneficial [166]; self-correction, experience, and workflow structure determine whether evidence improves a later decision [75, 78, 174]. Accordingly, fixed calls remain L1, evidence-adaptive computational tool use is L2, and autonomous physical execution with feedback is L3. The criterion is scientific authority exercised through tools, not the number of tools.

2.3.1 *Molecular state construction and knowledge grounding.*

Entity resolution and molecular state construction. A molecular agent first has to determine which chemical object it is acting on. User inputs may name a compound, give an IUPAC name, provide a CAS number, mention a target or protein family, quote a paper fragment, or describe an assay. The agent must resolve these surface forms into identifiers and representations such as SMILES, InChI, PubChem CID, ChEMBL ID, UniProt ID, PDB structures, or assay records. OPSIN, PubChem, ChEMBL, RCSB PDB, UniProt, ZINC, and Materials Project belong in this grounding layer [9, 49, 50, 55, 89, 142, 170]. A wrong molecule, target, protonation state, or assay condition can invalidate the rest of the workflow. ChemCrow [91] and CACTUS [93] rely on name-to-structure conversion and database grounding as entry points for tool use. ChemAgent [133], Chemist-X [16], and LLaMP [19] show how structured state construction can be combined with memory or retrieval.

Cheminformatics validation and descriptor computation. Cheminformatics toolkits provide many of the low-level operations that molecular agents need. RDKit [8] and Open Babel [101] parse SMILES and other formats, canonicalize structures, validate valence and aromaticity, preserve or check stereochemistry, compute descriptors, generate fingerprints, support similarity search, and convert file formats. These operations matter because LLM-generated molecular strings can be fragile, invalid, or underspecified. Once a text output becomes a checkable molecular object, the tool can act both as calculator and validator. ChemCrow [91] and CACTUS [93] use such operations for chemistry problem solving. MT-Mol [54], ToolMol [179], CheMatAgent [152], TRACE [75], and MolClaw [174] place them inside longer optimization or tool-planning loops. Evaluation can draw on established benchmarks for molecular design and property prediction, including MoleculeNet [154], GuacaMol [11], MOSES [110], and TDC [46].

Database and literature grounding. Molecular agents also need structured scientific memory. Databases and literature tools provide evidence about identity, target annotations, protein structures, bioactivity values, assay metadata, commercial availability, reaction precedent, and experimental results. Molecular databases differ from general web search because they preserve identifiers, units, curation history, and provenance. PubChem and ChEMBL ground compounds and bioactivity data; RCSB PDB and UniProt connect agents to protein structures and protein knowledge; ZINC and Materials Project support purchasable-molecule and materials retrieval [9, 49, 50, 55, 139, 142, 170]. Chemist-X [16] uses retrieved evidence for reaction-condition recommendation. LLaMP [19] grounds materials reasoning in high-fidelity retrieval, and agent-based literature learning extracts structured materials datasets from papers [3]. Retrieval-augmented drug-discovery agents extend this pattern to drug-target reasoning and collaborative molecular decisions [48, 62]. The difficult parts are entity linking, unit normalization, conflicting evidence, and versioned provenance.

2.3.2 *In-silico evaluation and simulation tools.*

Low-cost oracles and structure-based screening. In-silico evaluation tools let an agent test a hypothesis before synthesis or wet-lab work. Low-cost tools include descriptor calculators, drug-likeness filters, synthetic-accessibility scores, ADMET predictors, and toxicity predictors. Structure-based tools add docking, scoring, binding-pose analysis, and virtual screening. AutoDock Vina, GNINA, and DiffDock, together with benchmarks such as PDBbind and CASF, are common substrates for pose prediction, scoring, ranking, and screening [21, 94, 130, 140, 148]. These tools make iterative optimization practical: the agent proposes candidates, evaluates them with proxy objectives, and revises them. CACTUS [93] and MT-Mol [54] incorporate tool-based evaluation into agent workflows. Several drug-discovery agents combine generation, filtering, docking, retrieval, and optimization in more specialized settings [31, 61, 74, 125, 160, 161, 174, 179]. These scores should still be treated as decision signals, not as ground truth. Docking scores approximate binding, ADMET models depend on training distributions, and heuristic filters can reject unusual but useful chemistry.

Quantum chemistry and atomistic simulation. Quantum-chemical and atomistic simulation tools provide physical feedback at higher cost. They compute geometries, single-point energies, vibrational frequencies, reaction energetics, electronic structures, and material properties. ASE, xTB, Psi4, ORCA, Gaussian, pymatgen, and LAMMPS expose these calculations through programmable interfaces, which makes them usable inside agent workflows [5, 6, 29, 43, 96, 104, 107, 137]. For molecular agents, these tools ground part of the reasoning in physics rather than text or heuristic scores. El Agente [182] demonstrates autonomous quantum chemistry. ChemReasoner [128] uses quantum-chemical feedback to guide heuristic search over an LLM's chemistry knowledge space. ChemGraph exposes molecular simulation workflows as agent-compatible computational chemistry tasks [109]. Monte Carlo Thought Search [127] explores reasoning paths in catalyst design, and CheMatAgent [152] learns tool-use policies for chemistry and materials science. These systems reveal a demanding control problem: the agent must choose method, basis set, charge, spin state, solvent model, convergence

criterion, and computational budget. Materials-agent surveys make a similar argument for atomistic and materials workflows [171].

Molecular dynamics and free-energy workflows. Molecular dynamics tools move agents from static structures to time-dependent behavior. They support studies of protein flexibility, ligand stability, solvent effects, conformational transitions, and binding-related dynamics. GROMACS, OpenMM, AmberTools, and LAMMPS provide the computational substrate for many of these workflows [1, 15, 24, 137]. Free-energy workflows add methods such as FEP, TI, MM/PBSA, ABFE, umbrella sampling, and related protocols for ranking candidates. MDCrow [13] automates molecular-dynamics workflows, including setup, execution, and analysis. DynaMate [37] targets autonomous protein and protein-ligand MD workflows, while MDAgent2 [121] studies code generation, execution, evaluation, and self-correction for MD simulations. ToolMol [179], MADD [125], TRACE [75], modular task-execution agents [102], and MolClaw [174] combine MD or free-energy-like validation with docking, ADMET, and optimization. Evaluation has to consider stability, convergence, sampling sufficiency, cost, and whether simulation feedback changes downstream decisions.

2.3.3 Synthesis, characterization, and experimental action tools.

Reaction prediction and synthesis feasibility. A molecular candidate is not actionable until the agent can connect it to feasible synthesis, available starting materials, and executable reaction conditions. Reaction-prediction and retrosynthesis tools estimate whether a proposed molecule can be made, how many steps may be required, which reagents or catalysts are plausible, and whether route constraints match the intended application. AiZynthFinder and ASKCOS are useful reference points for this layer [33, 141]. Agentic systems increasingly treat synthesis planning as part of molecular design rather than as post-processing. ChemCrow [91] uses synthesis-oriented tools in chemistry problem solving. Chemist-X [16] focuses on reaction-condition recommendation, RETRO-R1 [86] studies agentic retrosynthesis, and Llamole [80] integrates inverse molecular design with retrosynthetic planning. ChemActor [177] converts synthesis procedures into structured chemical action sequences. Together, these works move the action space from attractive structures to candidates that can plausibly enter a Design, Make, Test, Analyze cycle.

Characterization, spectroscopy, and experimental readout. Closed-loop molecular agents must read experimental evidence, not just propose experiments. Characterization and spectroscopy tools convert measurements into structured feedback: NMR assignments, MS or LC-MS peaks, IR or UV spectra, XRD patterns, XANES features, microscopy images, plots, and assay readouts. The NIST Chemistry WebBook, MassBank, NMRShiftDB, and FDMNES provide representative infrastructure for thermochemical, spectral, NMR, MS, and XANES analysis [36, 44, 52, 59, 79, 97, 129]. This feedback tells the agent whether a reaction succeeded, whether the expected product formed, whether impurities appeared, and whether the next experiment should continue, change, or stop. LLM-RDF [116], ORGANA [22], and robotic ChemAgents [126] demonstrate experimental execution and readout interpretation. Tippy [27, 28] specifies corresponding laboratory-facing roles, but its reported evidence does not establish autonomous physical execution. ChemGraph-XANES [35] gives a concrete XANES simulation and analysis workflow for agents, while ChemLabs [156] examines multimodal reasoning in chemistry. The central issue is uncertainty: spectra and curves can support several hypotheses, so the agent should preserve alternatives instead of selecting one explanation too early.

Laboratory automation and safety gates. Laboratory automation tools move molecular agents from recommendation to physical action. They include protocol generators, robotic synthesis platforms, liquid-handling systems, cloud laboratories, reaction-execution interfaces, and instrument-control tools. This step creates stronger governance requirements. A failed computational job wastes time; a failed laboratory action can waste material, damage instruments, create unsafe conditions, or violate compliance rules. Coscientist [10], LLM-RDF [116], ORGANA [22], and robotic ChemAgents [126] combine execution, characterization, and feedback in physical workflows. AutoLabs [106] instead evaluates hardware-ready protocol generation with human verification before execution, while Tippy [27, 28] contributes a laboratory-facing architecture. Safety gates can include controlled-chemical checks, protocol validation, resource constraints, and human approval before high-risk actions, as suggested by safety-aware tool use in ChemCrow [91].

2.3.4 Tool orchestration and evaluation.

Workflow orchestration, state passing, and provenance. Molecular agency becomes most visible when tools are linked into workflows. An agent may pass a molecule from SMILES to SDF, convert a structure into docking input, prepare a PDB file for MD, summarize a trajectory into stability metrics, match spectra against candidate structures, or translate a reaction plan into a robot-executable protocol. Each transition can lose stereochemistry, protonation state, units, conformers, file-format details, or experimental conditions. ChemCrow [91] and CACTUS [93] demonstrate early chemistry tool orchestration. ChemHTS [77], ChemHAS [78], CheMatAgent [152], Mozi [14], and MolClaw [174] study more structured forms of tool stacking, tool learning, hierarchical skills, or governed autonomy. Longer-horizon systems such as El Agente [182], ChemGraph [109], ChemGraph-XANES [35], MDCrow [13], DynaMate [37], MDAgent2 [121], and LLaMP [19] point to the need for state tracking, logging, versioning, intermediate-memory management, and provenance records. ToolUniverse points in the same direction for general scientific tooling, where standardized specifications and reusable composition become infrastructure [32].

Failure recovery, cost control, and calibration. Tool use also introduces failure modes that are specific to chemistry. A workflow can fail because of invalid SMILES, lost stereochemistry, missing hydrogens, inconsistent protonation states, malformed docking inputs, DFT non-convergence, unstable MD trajectories, database mismatches, unit errors, unavailable reagents, or unsafe protocols. A robust molecular agent should detect these failures, explain them, retry with corrected inputs, switch tools when appropriate, or request human intervention. Tooling-or-Not-Tooling [166] motivates this caution by showing that tools can be harmful when

used in the wrong setting. ChemHAS [78], TRACE [75], and MolClaw [174] suggest that self-correction, action-level experience, and workflow-level skills can improve tool-facing behavior. Computational workflow agents such as El Agente [182], ChemGraph [109], MDCrow [13], DynaMate [37], and MDAgent2 [121] make cost control especially important, since high-cost DFT, MD, or free-energy calculations should not run when a lower-fidelity check is enough. Calibration matters for the same reason. The agent should know when a proxy score is useful for ranking, when higher-fidelity validation is needed, and when the evidence remains inconclusive.

Evaluation across the autonomy ladder. Toolbox evaluation should retain a tool-specific view even when the detailed benchmark taxonomy is consolidated in Section 3. Beyond final-answer accuracy, it should test molecular identity preservation, valid calls, state passing, recovery, provenance, and whether tool evidence improves a later decision. MolViBench [70], MolBench [174], and MatTools [81] expose parts of this process, while L3 evaluation additionally requires protocol executability, hardware compatibility, safety, and faithful interpretation of measurements.

2.4 Learning and Optimization

Learning and optimization describe two complementary dimensions of improvement in molecular agents. **Learning concerns what the agent acquires or updates from experience**, including policies, memories, predictive models, acquisition strategies, tool-use behaviors, and reusable workflows. **Optimization concerns what component or decision variable is deliberately improved with respect to an objective**, ranging from molecular candidates and experimental conditions to action sequences, tool choices, search policies, and the workflow itself.

The distinction is therefore not simply between improvement within a task and improvement across trajectories. Optimization may operate within a single trajectory or across many episodes, while learning may occur online during a trajectory or accumulate across repeated tasks and campaigns. The two processes also interact: optimization can exploit previously learned knowledge, while learning itself is often driven by an optimization objective.

For molecular agents, the important question is not whether a system contains a learning algorithm or repeatedly performs optimization. Rather, the evidence for agency lies in whether observations, evaluations, or accumulated experience change a later consequential scientific decision. The autonomy level then depends on *what kind of decision is changed*: a fixed computational procedure remains L1; computational evidence that changes a later candidate, plan, tool call, or stopping decision supports L2; physical execution with incorporated experimental feedback supports L3; and accumulated evidence that changes objectives, strategy, or scientific agenda across campaigns is required for L4.

2.4.1 Optimization Targets and Search Spaces. Early molecular learning and optimization systems focused primarily on candidate-level decisions, such as generating a molecule or editing a molecular graph. More recent molecular agents broaden the object of optimization from the molecular candidate to the discovery process itself. They may optimize which candidate to evaluate, which tool to invoke, which evidence to trust, how computational resources are allocated, when a failed action should be repaired, and whether a multi-step trajectory should continue or stop.

Section 2.1 describes molecular representations in detail; here, the relevant issue is the candidate and action spaces they expose to an optimizer. Sequence policies generate or edit strings, whereas graph policies act on atoms, bonds, fragments, or scaffolds. Property-directed SMILES generation in Molecular De Novo Design [103] and ReLeaSE [111] established the propose–score–update pattern. REINVENT4 extends this paradigm through transfer learning, reinforcement learning, curriculum learning, and multi-component scoring [88].

String actions integrate readily with language models but can fail at ring closures, stereochemistry, or syntax. SELFIES, Group SELFIES, and SAFE reduce some syntactic failures through constraints or chemically meaningful units [18, 58, 100], although validity does not imply stability or synthesizability. Graph actions instead make molecular topology and local edits explicit. GCPN constructs graphs under validity constraints, whereas MolDQN performs local lead-optimization edits from an existing molecule [165, 181].

Representation determines what can be changed, but it does not determine whether the system is agentic. The same optimization policy may appear inside a fixed human-controlled pipeline or an adaptive agent. What matters is whether feedback from an earlier action alters a later proposal, evaluation strategy, tool choice, or selection decision.

2.4.2 Multi-Objective Decision Making. Evaluation provides the objectives and constraints under which optimization proceeds. Early molecular optimization studies often emphasized a single objective, such as QED, LogP, molecular similarity, predicted binding affinity, or docking score. Drug discovery and materials design, however, generally require simultaneous consideration of activity, selectivity, toxicity, ADMET, stability, synthetic accessibility, novelty, diversity, and cost.

Because these criteria may conflict, optimization commonly relies on composite objectives, constrained optimization, threshold-based filtering, Pareto selection, or combinations of these mechanisms. DrugEx v2 [87] combines reinforcement learning with Pareto ranking to balance multiple molecular properties rather than maximize a single weighted score. QADD [180] performs iterative multi-objective reinforcement learning for de novo drug design, while MARS [155] combines Markov chain Monte Carlo with graph editing in a propose–evaluate–retain loop.

Multi-objective decision making is more than assigning a score. It determines which trade-offs are acceptable, which constraints are non-negotiable, and which candidates receive additional computational or experimental resources. Some objectives can be balanced against one another, whereas synthesis feasibility, safety, or executability may instead be enforced as hard constraints.

These mechanisms remain optimization methods rather than evidence of autonomy by themselves. They contribute to an L2 agent only when their evaluations affect a subsequent candidate, edit, tool invocation, resource-allocation decision, or stopping condition without an intervening human decision. Likewise, applying an optimizer to an experimental objective does not establish L3 unless physical execution and incorporated experimental feedback are demonstrated.

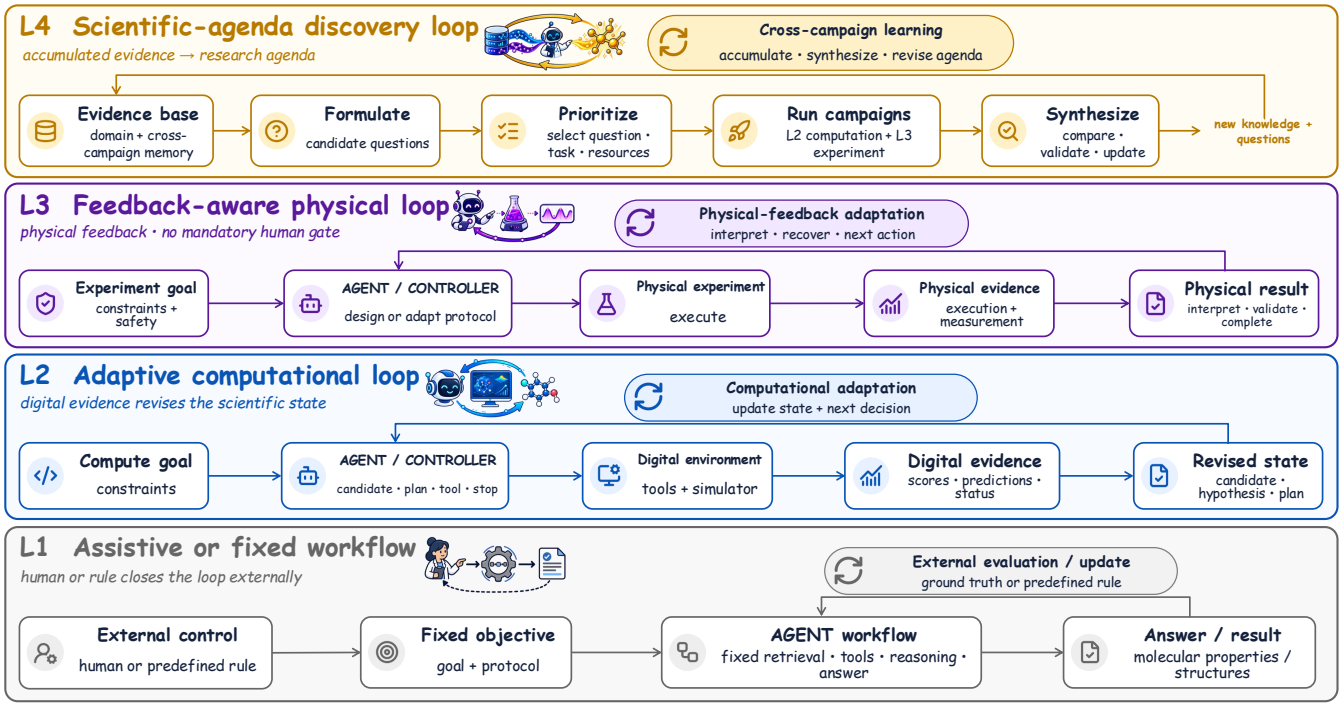

Figure 4: Hierarchical learning and optimization in molecular LLM agents. Optimization may target candidates, actions, workflows, or experiments, while learning updates policies, memories, models, strategies, and reusable workflows from experience. These mechanisms can operate at multiple autonomy levels: from fixed computational procedures (L1), through feedback-driven computational decisions (L2) and physical experimentation (L3), to cross-campaign learning that revises higher-level scientific objectives or agendas (L4).

Most computational optimization relies on surrogate objectives, including property predictors, docking scores, estimated binding affinity, synthetic accessibility, ADMET models, and other low-cost approximations. These signals make repeated evaluation feasible, but they are not equivalent to the scientific outcomes ultimately sought.

A sufficiently capable optimizer may therefore exploit weaknesses in a proxy rather than discover a scientifically useful candidate. Such failure becomes especially consequential in long-running agentic loops because biased feedback can affect many subsequent decisions. Robust optimization should consequently incorporate validity checks, uncertainty estimates, applicability-domain analysis, diversity constraints, and higher-fidelity confirmation where appropriate, while explicitly distinguishing predicted evidence from measured evidence.

2.4.3 Exploration and Acquisition. Optimization also requires deciding where additional search or evidence is most valuable. Chemical space is extremely large, and useful regions are sparse. An optimizer that exploits current rewards too aggressively may converge around a narrow family of high-scoring structures and repeatedly make minor local modifications, reducing scaffold diversity and increasing vulnerability to biased surrogate objectives.

Mol-AIR [159] introduces adaptive intrinsic rewards for goal-directed molecular generation, encouraging novelty and broader coverage in addition to target-property performance. Augmented Hill-Climb [136] improves the sample efficiency of REINVENT-like molecular language models, reducing the number of unproductive candidates requiring downstream evaluation.

When evaluations are expensive, acquisition itself becomes an optimization problem. Phoenics [40] uses previous observations and predictive uncertainty to recommend candidates or conditions expected to provide either high objective value or useful information. The constrained latent-space approach of Gómez-Bombarelli et al. [34] similarly provides a smoother domain in which to conduct search than direct optimization over discrete strings or graphs.

Exploration becomes agentic when accumulated evidence changes where the controller searches, what information it chooses to acquire, how much resource it allocates to an evaluation, or when it decides that further search is no longer worthwhile. This distinction becomes increasingly important as the workflow moves from inexpensive computational scoring toward simulations and physical experiments.

2.4.4 Agent Learning and Workflow Optimization. Molecular agents extend improvement beyond the candidate itself. An agent may *learn* which tools are reliable, which representations are useful, which failures recur, which search regions are productive, or which workflows succeed under particular objectives. It may then *optimize* later behavior using this acquired information by changing

its policy, tool sequence, resource allocation, repair strategy, or stopping rule.

ReMol [150] combines LLM guidance with reinforcement learning. The language model contributes chemical priors and reasoning signals, while reinforcement learning updates the molecular policy using property feedback. This illustrates how learned knowledge can influence optimization without restricting the LLM to direct molecular-string generation.

ChemCRAFT [64] learns tool-use policies from trajectories in chemical sandboxes. MolClaw [174] organizes tool-level, workflow-level, and discipline-level skills for molecular evaluation, screening, and optimization. These systems illustrate learning at the level of agent behavior rather than only at the level of molecular generation.

General agent-learning methods provide related mechanisms. Reflexion [123] stores natural-language feedback that can alter behavior in later attempts without parameter updates. Agent Lightning [90] represents multi-step agent execution as a Markov decision process and applies reinforcement learning to assign credit across trajectories.

The existence of memory, reinforcement learning, or trajectory storage nevertheless does not establish scientific agency. Learning is scientifically consequential only when acquired information is reused to change a later decision, and when the effect is demonstrated beyond the episode from which the information was obtained. Likewise, workflow optimization requires evidence that feedback changes a consequential candidate, hypothesis, plan, tool choice, resource-allocation decision, or stopping rule. Merely executing a predefined sequence of tools, storing a transcript, or correcting tool syntax remains compatible with L1.

2.4.5 Laboratory-in-the-Loop Optimization. Laboratory-in-the-loop optimization replaces or supplements computational proxies with physical observations. The optimization target may be a molecular candidate, reaction condition, formulation, synthesis protocol, or experimental sequence, while learning can update the acquisition policy, predictive model, experimental strategy, or workflow from observed outcomes.

Experimental feedback is typically slower, noisier, and more expensive than computational scoring. The agent must therefore balance target performance with information gain, material use, executability, reproducibility, uncertainty, and safety.

Bayesian reaction optimization [122] demonstrates how previous observations and uncertainty can guide the selection of subsequent experimental conditions. The method provides an experimental learning and optimization mechanism, although that mechanism alone does not establish an LLM-centered autonomous agent.

Self-driving laboratories provide clearer evidence of closed physical loops. AlphaFlow [146] uses reinforcement learning to guide a microfluidic platform in the exploration and optimization of multi-step chemical processes. Autonomous Polymer Synthesis [57] performs multi-objective closed-loop optimization for polymer synthesis under Pareto trade-offs. In both cases, measured outcomes affect subsequent experimental decisions.

LLM-centered systems increasingly connect planning with laboratory execution. Coscientist [10] plans and executes experiments through equipment or cloud-laboratory interfaces and uses feedback to interpret measurements or repair execution code. ORGANA [22] combines agent-generated plans with visual feedback during physical execution. AutoLabs [106] evaluates self-corrected, hardware-ready protocol generation, but a mandatory human verification step before execution prevents the reported workflow from satisfying L3 under our criterion.

A stronger form of L3 occurs when a measurement determines another experiment. LLM-RDF [116] uses measured reaction yields to select subsequent reaction conditions. Robotic ChemAgents [126] uses measured catalyst performance to select later compositions for physical validation. The Mobile Robotic Chemist [12] and A-Lab [131] likewise use observed experimental outcomes to choose subsequent physical actions.

These cases distinguish broad L3 from iterative L3. Broad L3 requires autonomous physical execution together with incorporated execution or measurement feedback that changes recovery, interpretation, or completion decisions. Iterative L3 requires the stronger condition that a measured result determines a subsequent physical experiment or experimental condition.

Human monitoring and emergency stopping do not necessarily reduce the autonomy level, whereas a mandatory approval gate before routine physical execution does. Fixed protocol replay remains L1, and a proposed laboratory architecture without demonstrated physical execution does not establish L3.

2.4.6 Cross-Trajectory and Cross-Campaign Learning. Learning can persist beyond a single trajectory. Across repeated tasks, an agent may accumulate reusable memories, update policies, estimate tool reliability, refine acquisition strategies, or induce workflow-level skills from previous successes and failures. Such cross-trajectory learning can improve later L2 or L3 decisions without necessarily changing the scientific objective itself.

Cross-campaign learning is stronger. Here, accumulated evidence from completed optimization or experimental campaigns alters higher-level scientific choices, such as which objective to pursue, which hypothesis to investigate, which region of chemical space deserves further study, which experimental strategy should be abandoned, or how future campaigns should be organized.

This distinction is important for L4. Reusing a successful workflow, updating a policy from previous trajectories, or fine-tuning an agent from accumulated experience may constitute learning, but it does not by itself establish L4 autonomy. L4 requires evidence that learning changes a consequential scientific decision at the campaign or agenda level rather than merely improving execution of a previously specified objective.

2.4.7 What Learning and Optimization Evidence Establishes. Learning and optimization mechanisms should therefore not be mapped directly onto autonomy levels. Reinforcement learning, Bayesian optimization, memory, multi-objective selection, and workflow adaptation can all occur at different levels depending on how their outputs affect subsequent decisions.

A fixed generator, scorer, optimizer, or workflow remains L1 when iteration is prescribed by a human or fixed procedure. L2 requires computational evidence or learned experience to change a later candidate, hypothesis, plan, tool call, acquisition decision, or stopping condition. L3 additionally requires physical execution together with incorporated execution or measurement feedback;

Table 2: Landscape of representative molecular and scientific-agent benchmarks. The table emphasizes evaluated interaction and evidence rather than ranking benchmark-specific scores, which are not directly comparable.

Benchmark	Scientific scope	Agent interaction or loop	Primary evaluation signal
MolViBench [70]	358 molecular tasks, 12 workflows, and five difficulty levels	Generates executable programs for multi-step molecular workflows	Program correctness, execution success, and degradation with workflow complexity
MolBench [174]	Molecular screening, optimization, and end-to-end discovery challenges spanning 8–50+ tool calls	Connects filtering, affinity estimation, molecular editing, and workflow execution	Subtask quality and end-to-end challenge completion
ChemCost [153]	1,427 reactions, 2,261 chemicals, and 230,775 price quotes	Reasons over reaction components and noisy procurement information	Cost-error tolerance, robustness to noise, and component-level attribution
MDGym [60]	169 molecular simulations and 303 tasks across two MD engines	Configures, runs, diagnoses, and repairs simulation workflows	Executable task success by difficulty, engine, and failure type
ChemReason-Bench [172]	7,306 tasks instantiated from 500 organic reactions in six formats	Produces or validates ordered, condition-aware, schema-constrained procedure steps	Ordering, constraint validation, entity-role grounding, and parseable completion
Corral [2]	Four chemistry and materials environments: MD, ML, catalysis, and spectroscopy	Compares tool-calling and ReAct-style agents in executable expert-designed tasks	Task success, tool-use failures, and sensitivity to task–tool alignment
MADE [92]	Closed-loop computational materials discovery over chemical systems	Proposes and evaluates candidates under a constrained oracle budget, then adapts the search	Discovery efficacy, efficiency, and scaling with search-space complexity
ScienceAgentBench [17]	102 tasks derived from 44 papers across four disciplines	Produces executable research code with paper and data context	Full-task and partial success, expert-knowledge dependence, and execution correctness
SciAgentBench [120]	259 tasks, 1,134 subquestions, and 1,780 scientific tools	Selects and composes tools from elementary calls to long workflows	Step and overall success, tool routing, and performance versus interaction horizon
SciCode [138]	80 scientific coding problems and 338 expert-designed subproblems	Implements research algorithms from specifications and intermediate requirements	Subproblem and full-problem execution success
CORE-Bench [124]	270 reproducibility tasks from 90 papers in three disciplines	Reproduces published results from code, data, text, and visual artifacts	Reproduction accuracy across difficulty levels and modalities
SciAgentArena [85]	Approximately 200 real-world scientific tasks across multiple domains	Solves interactive research scenarios with stepwise verification	Stepwise task completion and behavior in specified versus open-ended scenarios

measurement-selected follow-up experiments provide the stronger iterative form. L4 requires learning across campaigns to influence higher-level objective selection, strategy, hypothesis formation, or scientific agenda.

Current evidence therefore spans candidate and workflow optimization at L1 and L2, together with bounded physical optimization and learning at L3. Coscientist and ORGANA [10, 22] demonstrate feedback-aware physical workflows, while AlphaFlow, autonomous polymer synthesis, LLM-RDF, Robotic ChemAgents, the Mobile Robotic Chemist, and A-Lab [12, 57, 116, 126, 131, 146] demonstrate measurement-driven experimental iteration. These results show increasingly capable learning and optimization loops, but none by itself establishes the cross-campaign objective selection and scientific-agenda revision required for L4.

3 Evaluation and Benchmarking

Evaluation must distinguish component competence from scientific loop closure. Executable-workflow benchmarks can reveal whether an agent constructs valid programs, selects tools, and completes multi-step interactions correctly [70, 120]. Closed-loop discovery environments ask a different question: whether observations redirect a budgeted search toward better candidates [92]. A chemistry question-answering score therefore cannot establish reliable tool use, while a single end-to-end success rate can hide whether the decisive contribution came from the controller, foundation model, tool, or evaluator. We organize representative benchmarks by the interaction they expose and the scientific state they require the agent to change.

Because benchmark-specific scores are not on a common scale, Figure 5 provides one compact, source-faithful result slice for each benchmark in Table 2, rather than a cross-benchmark aggregate. Molecular-dynamics or chemistry subsets are used when a primary source tabulates them. ScienceAgentBench and SciCode include computational chemistry or chemistry problems but do not tabulate model-level chemistry scores, and CORE-Bench has no molecular split; their panels are therefore explicitly marked as overall results. Each panel retains its source metric and must be interpreted locally.

Figure 5 supports four observations when the scores are read together with each benchmark's design, rather than as a shared leaderboard.

Insight 1: benchmark scores answer different scientific questions. ChemReason-Bench and ChemCost evaluate bounded procedural or cost reasoning against specified targets [153, 172]. MolViBench, MDGym, Corral, and SciAgentBench instead require executable tool sequences, making routing, intermediate validity, and recovery part of the evaluated object [2, 60, 70, 120]. ScienceAgentBench, SciCode, and CORE-Bench extend the horizon to research code or reproducibility artifacts, whereas MADE and the SciAgentArena optimization slice evaluate feedback-guided search under an oracle budget [17, 85, 92, 124, 138]. Thus, superficially similar percentages can denote answer accuracy, executable completion, reproduction success, or search efficiency.

Insight 2: outcomes are configuration-sensitive, but more tools are not uniformly better. In the reported MolBench slice, MolClaw-CC reaches 81.1% accuracy versus 51.4% for vanilla agents and a 45.1% standalone-LLM mean. For matched ChemCost backbones, ReAct with tools is associated with a CTA@25 change from 2.5–4.1% to 34.6–50.6% [153, 174]. Yet SciAgentBench's OtherTools ablation scores 21.0% versus 28.6% for the base configuration, and Corral changes the relative ordering of ReAct and tool calling across backbones and single versus chained tasks [2, 120]. Framework claims therefore need matched backbone, controller, tool-set, and task conditions.

Insight 3: longer or harder execution exposes a reliability gap. SciCode reports 21.2–28.5% subproblem Pass@1 but only 1.5–7.7% on complete main problems [138]. MDGym full success falls from at most 21% on easy tasks to at most 4% on hard tasks, while CORE-Bench drops from 60.0% to 21.5% for GPT-4o and from 44.4% to

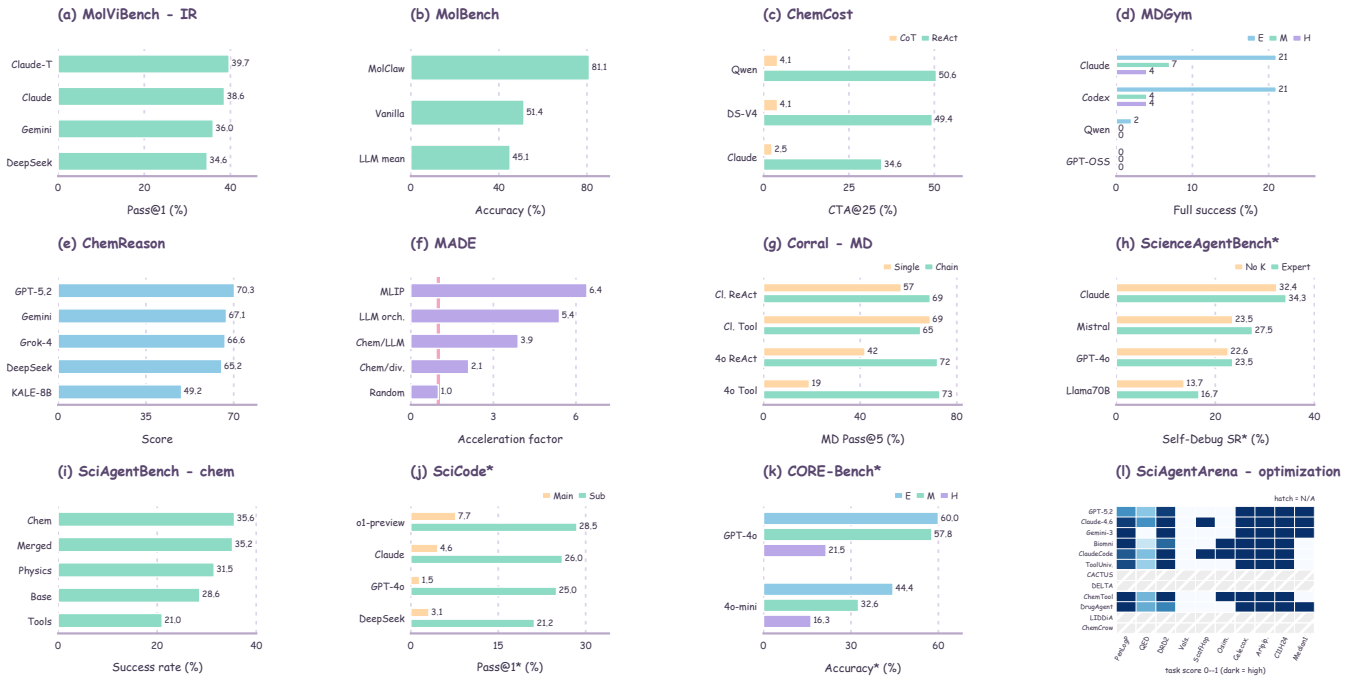

Figure 5: Compact reported result slices for all 12 benchmarks in Table 2: (a) MolViBench IR [70]; (b) MolBench [174]; (c) ChemCost [153]; (d) MDGym [60]; (e) ChemReason-Bench [172]; (f) MADE [92]; (g) Corral MD [2]; (h) ScienceAgentBench [17]; (i) SciAgentBench chemistry [120]; (j) SciCode [138]; (k) CORE-Bench [124]; and (l) SciAgentArena molecule optimization [85]. Asterisks mark overall results used where no numeric molecular subgroup is published. In (l), darker cells denote higher task scores on the source's 0–1 scale and hatching denotes incompatible agent–task combinations. Axes retain source metrics and are comparable only within panels.

16.3% for GPT-4o-mini [60, 124]. Across these distinct designs, the pattern is consistent with errors accumulating across specification, execution, diagnosis, and repair. Partial credit is informative, but should be paired with end-to-end success and results stratified by horizon or difficulty.

Insight 4: the metric must expose efficiency and coverage, not only success. MADE reports acceleration over a random baseline under a fixed oracle budget, so it measures how efficiently feedback redirects discovery rather than whether a single answer is correct [92]. ScienceAgentBench separately reports partial/full success and expert-knowledge conditions, revealing that added context does not remove the execution gap in the displayed settings [17]. SciAgentArena's heatmap further combines strong, failed, and unsupported agent–task pairs [85]; a mean over completed cells would therefore hide both specialization and missing coverage.

These characteristics also delimit autonomy claims. L1 can use bounded correctness or fixed-workflow completion, whereas L2 needs evidence that an observation changes a candidate, plan, tool choice, or stopping decision, along with loop completion, recovery, cost, and oracle use. L3 additionally requires repeatable physical execution, measurement-conditioned decisions, human interventions, and safety violations, consistent with multidimensional self-driving-laboratory evaluation [145]. No benchmark in Table 2 yet establishes a general L3 standard or the cross-campaign agenda revision required for L4. Accordingly, headline scores should be accompanied by task coverage, intermediate validity, end-to-end success, efficiency, reproducibility, operational domain, and mandatory human gates.

4 Evidence for Scientific Autonomy Levels

Molecular LLM agents differ in their scientific capabilities and in the extent to which they independently control a scientific workflow. Laboratory-autonomy frameworks separate physical process execution from data analysis, interpretation, decision-making, and communication, and they classify systems partly by the decisions that still require a human researcher [47, 145]. We follow this emphasis on decision authority while adapting it to LLM-centered molecular agents. Implementation features such as planning, memory, reinforcement learning, tool use, or multi-agent coordination remain enabling mechanisms: a multi-agent system may require approval at every consequential step, whereas one controller may complete a narrow experimental episode without such a gate.

As shown in Figure 2, we define scientific autonomy according to the *outermost feedback loop that an agent can reliably close without mandatory human intervention*. This is an operational rubric for this survey rather than a claim that the field has converged on one universal scale. The distinction concerns decision responsibility and the source of feedback, not the number of tools, reasoning steps, model components, or repeated trials.

Scientific workflows as nested feedback loops. Let τ denote one scientific workflow episode executed by agent A. We consider three nested forms of loop closure:

$$d \in \{\text{comp}, \text{phys}, \text{sci}\}, \tag{1}$$

corresponding respectively to computational, experimental, and scientific-agenda loops.

For each loop type d, define

$$C_d(A, \tau) = \mathbb{I}\left[\text{Loop}_d(A, \tau) \wedge H_d(\tau) = 0\right], \tag{2}$$

where $\text{Loop}_d(A, \tau)$ indicates that the corresponding feedback loop is completed, $H_d(\tau)$ is the number of mandatory human decision gates inside that loop, and $\mathbb{I}[\cdot]$ is the indicator function.

A mandatory human gate is an intervention without which the ordinary workflow cannot continue. Human monitoring, retrospective inspection, emergency stopping, and specification of initial constraints do not count as mandatory gates when the agent can otherwise proceed. By contrast, required protocol verification before instrument execution is a gate; AutoLabs explicitly reports such a verification step [106].

The three indicators are defined as follows.

- $C_{\text{comp}} = 1$ if the agent uses tool or environment evidence to revise a scientifically consequential state or decision, such as a candidate, hypothesis, plan, tool choice, or stopping rule, without an intermediate human decision. When no physical experiment is controlled, this evidence must come from a digital or simulated environment. Executing a fixed pipeline or repairing only syntax does not satisfy this criterion.
- $C_{\text{phys}} = 1$ if $C_{\text{comp}} = 1$ and the agent additionally designs or adapts and executes a physical experiment and incorporates execution or measurement feedback into recovery, state, interpretation, or completion without an intermediate human decision. Selecting a subsequent experiment from measurements is a stronger iterative form of physical autonomy, but is not required for $C_{\text{phys}} = 1$.
- $C_{\text{sci}} = 1$ if $C_{\text{phys}} = 1$ and the agent additionally uses accumulated evidence to formulate, prioritize, and pursue new scientific questions, including the autonomous selection of transitions across different scientific task families.

These loop types are hierarchical:

$$C_{\text{sci}}(A, \tau) \le C_{\text{phys}}(A, \tau) \le C_{\text{comp}}(A, \tau). \tag{3}$$

Scientific-agenda autonomy therefore presupposes the ability to manage the experimental and computational processes required to investigate selected questions. Physical autonomy likewise presupposes adaptive planning, analysis, or control. A robot that only replays a fixed protocol does not satisfy either C_{comp} or C_{phys}. This criterion separates automation of execution from adaptive decision-making, a distinction also made in laboratory-autonomy frameworks [47].

Autonomy-level definition. Given the nested loop indicators, the episode-level autonomy of agent A is defined as

$$L(A, \tau) = 1 + C_{\text{comp}}(A, \tau) + C_{\text{phys}}(A, \tau) + C_{\text{sci}}(A, \tau). \tag{4}$$

Because of Eq. (3), $L(A, \tau) \in \{1, 2, 3, 4\}$.

This produces the following four levels.

Table 3: Levels of scientific autonomy for molecular LLM agents.

Level	Closed loop	Operational criterion
L1	None	The agent retrieves information, invokes tools, provides recommendations, or executes a fixed workflow, but evidence does not autonomously revise a consequential scientific decision.
L2	Adaptive computational	Given a computational objective, the agent uses digital evidence to revise a candidate, hypothesis, plan, tool choice, or stopping decision within a digital or simulated environment.
L3	Physical workflow	Given a high-level experimental objective, the agent designs and executes a physical experiment and incorporates execution or measurement feedback without a mandatory human gate during ordinary operation.
L4	Scientific agenda	The agent formulates and prioritizes new scientific questions, selects suitable computational and experimental task families, and updates its research agenda from accumulated evidence.

The four levels may therefore be summarized as follows: L1 systems remain assistive or fixed; L2 systems adapt a computational scientific state from digital evidence; L3 systems autonomously complete a feedback-aware physical workflow; and L4 systems additionally control the evolution of the scientific agenda.

Clarifying the level boundaries. The number of tool calls or agents is not an autonomy criterion. A workflow involving many tools remains at L1 when its sequence is fixed or when outputs do not change a later scientific decision. A relatively simple optimization workflow may qualify as L2 if computational evidence changes the candidate, plan, tool choice, or stopping rule. ReAct-style task completion, format repair, and review-only debate do not qualify by themselves.

Physical execution alone is also insufficient for L3. A robotic platform that replays a fixed human-written protocol performs automation. L3 requires the agent to formulate or adapt the experimental procedure and to use physical feedback in recovery, interpretation, or completion.

The statement that L3 *does not require a second experiment* refers only to the number of physical trials, not to the absence of feedback. A single episode can qualify when an observation changes what the agent does or concludes inside that episode. For example, Coscientist reads UV–Vis spectra to identify an initially unknown physical state and, in a separate integrated experiment, consults hardware documentation and repairs an invalid automation method before successful execution [10]. ORGANA likewise uses visual feedback to guide long-horizon physical plans [22]. Neither example needs a second synthesis to demonstrate feedback-aware physical control. When a measured yield or material property selects another experiment, as in LLM-RDF and robotic ChemAgents [116, 126], we label the evidence *iterative L3*. This distinction is specific to our physical-workflow rubric; SDL taxonomies often reserve their stronger closed-loop categories for systems that also automate experiment selection across repeated trials [145].

Generating candidate hypotheses is also insufficient for L4. The agent must evaluate candidate questions, select one to pursue, allocate computational or experimental actions, and revise its research

Table 4: Representative molecular-agent systems classified by the strongest feedback loop demonstrated in their reported evaluation. The examples are illustrative rather than exhaustive; level is assigned from evidence-conditioned behavior, not from the number of tools or agents.

System	Scientific setting	Strongest demonstrated feedback or action	Assigned level
ChemAgent [133]	Chemical reasoning	Updates task memory to improve bounded reasoning, without closing an external scientific feedback loop	L1: assistive memory
CACTUS [93]	Chemistry tool use	Executes a predefined tool-mediated problem-solving workflow	L1: fixed workflow
Chemist-X [16]	Reaction conditions	Retrieves evidence and recommends conditions; downstream experimental revision remains external	L1: recommendation
ChemCrow [91]	General chemistry	Uses computational chemistry tools and their outputs to revise a tool-mediated solution trajectory	L2: computational loop
ChatDrug [84]	Molecular editing	Uses retrieval and domain feedback to revise subsequent molecular edits	L2: feedback-driven edit
ChemReasoner [128]	Catalyst and molecular search	Uses quantum-chemical rewards to redirect search over candidate reasoning paths	L2: simulation feedback
MDCrow [13]	Molecular dynamics	Uses execution and analysis feedback to diagnose and repair computational MD workflows	L2: workflow adaptation
TRACE [75]	Lead optimization	Reuses tool trajectories and changes candidates or tool plans from computed evidence	L2: optimization loop
Coscientist [10]	Chemical experimentation	Plans and executes physical experiments and incorporates execution feedback during ordinary operation	L3: physical workflow
LLM-RDF [116]	Robotic chemistry	Converts plans into robotic execution and interprets experimental readouts	L3: experiment feedback
ORGANA [22]	Automated laboratory	Generates procedures, executes them physically, and uses observed results in completion decisions	L3: experiment feedback
AutoLabs [106]	Laboratory protocol generation	Self-corrects hardware-ready procedures, followed by required human verification before execution	L2: computational workflow
No surveyed system	Cross-campaign discovery	No evaluated system demonstrates evidence-conditioned formulation and revision of a scientific agenda across task families	L4: evidence gap

direction from the evidence. Transitions between task families must be evidence-conditioned rather than fixed entirely in advance.

Domain-conditioned autonomy. An autonomy claim is meaningful only relative to the domain in which the system operates; existing laboratory frameworks likewise caution that a level does not by itself define the scope of the research being categorized [47]. Let $\mathcal{D}$ denote a scientific operational domain specifying supported tasks, tools, environments, instruments, resources, and safety constraints.

The system-level autonomy within $\mathcal{D}$ can be defined as

$$L_{\mathcal{D}}(A) = \max\left\{\ell : \Pr_{\tau\sim\mathcal{D}}\left[L(A,\tau) \geq \ell\right] \geq \rho\right\}, \tag{5}$$

where ρ is a predefined reliability threshold.

Safety can be imposed as a separate evaluation constraint:

$$\Pr_{\tau\sim\mathcal{D}}\left[U(A,\tau) = 1\right] \leq \delta, \tag{6}$$

where $U(A,\tau)$ indicates an unsafe or invalid episode and δ denotes the maximum acceptable violation rate.

Separating Eqs. (5) and (6) is important because autonomy and competence are distinct. A system may be authorized to complete an L3 workflow but do so unreliably, whereas a highly accurate molecular predictor may remain at L1 because it neither determines nor executes subsequent actions.

Complementary continuous measures. The discrete level can be accompanied by continuous measures of human involvement and loop reliability. This follows proposals to report autonomy alongside lifetime, throughput, precision, material use, accessible parameter space, and optimization performance [145]. For example, the mandatory human-intervention rate may be reported as

$$R_{\mathrm{H}} = \frac{N_{\mathrm{gate}}}{N_{\mathrm{decision}}}, \tag{7}$$

where N_{gate} is the number of mandatory human gates and N_{decision} is the number of consequential workflow decisions.

For loop type d, the empirical loop-completion rate is

$$R_d = \frac{1}{N}\sum_{i=1}^{N} C_d(A,\tau_i). \tag{8}$$

These measures distinguish systems that nominally belong to the same autonomy level but differ substantially in robustness or dependence on human intervention.

Relationship to enabling technologies. Planning, memory, reinforcement learning, and multi-agent coordination are orthogonal to the autonomy levels. They are enabling mechanisms whose roles change across levels rather than level-defining properties.

At L1, memory may support information retrieval and personalized assistance. At L2, it may preserve state across long computational workflows and support recovery from failed tool calls. At L3, it may maintain experimental context across physical iterations. At L4, it may organize evidence accumulated over multiple projects and support long-term research-agenda formation.

The same principle applies to reinforcement learning and multi-agent coordination. Their contribution to autonomy should be assessed by whether they enable the system to close a broader feedback loop, rather than by their mere presence in the architecture.

5 Safety and Challenges

The architectural components in Section 2 expose distinct but coupled failure modes. Perception can corrupt molecular identity, the agent core can select an unsupported action, a tool can return unreliable evidence, and optimization can amplify weaknesses in any of the preceding components. In this section, we provide an overview of the safety concerns and then organize the open challenges along the same four-part architecture which can be summarized as Figure 6.

5.1 Safety and Governance Boundaries

The risk posed by a molecular agent is determined by its operational authority, the reversibility of its actions, the scope of substances and equipment it can affect, and the consequences of an erroneous decision. The self-driving-laboratory safety literature accordingly treats safety as a property of interactions among software, hardware, materials, people, and operating procedures rather than of the language model alone [63]. For instance, read-only retrieval and identity checks can be assigned to a control surface narrower

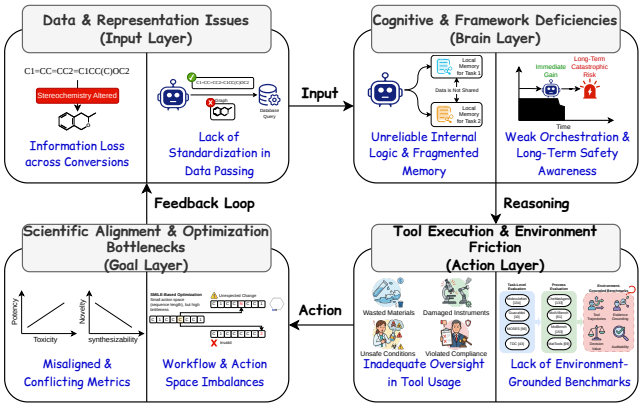

Figure 6: Challenges and bottlenecks in the lifecycle of a molecular agent. The diagram maps vulnerabilities across perception, controller design, tool and environment interaction, and optimization; arrows indicate how an upstream error can propagate into more consequential system-level behavior as action authority increases.

than purchasing, synthesis planning, or instrument control. Furthermore, the same tool may also require different permissions across operational domains.

The safety concerns raised by molecular discovery agents can be classified into three broad categories: intrinsic system limitations, extrinsic human-induced threats, and physical-world operational risks.

At the intrinsic level, these agents rely on reasoning, planning, and memory modules that are prone to instability or insufficiency, which can lead to hallucinations or erroneous predictions of molecular properties. Such inaccuracies introduced during early-stage information retrieval can cascade into catastrophic failures through synthesis planning and robotic execution [14, 41]. When such erroneous information guides experimental workflows, it can resulting in resource waste, generation of hazardous byproducts, and potential laboratory incidents. Moreover, large language models are known to struggle with long-horizon planning and complex reasoning tasks, impairing the agent's ability to anticipate downstream consequences or recognize critical safety checkpoints in multi-step synthetic routes [134].

On the human–agent interface, extrinsic safety is predominantly challenged by the diversity of user intent. Instructions may range from benign requests to deliberately malicious prompts. Adversarial inputs and jailbreak techniques can subvert the agent's safety alignment, compelling it to propose synthetic pathways for hazardous compounds. Dual use is a concrete molecular design concern, the same capabilities that enable beneficial drug discovery can be repurposed to design highly toxic compounds, controlled substances, or chemical weapons [143].

However, the most consequential safety challenge emerges when molecular discovery agents are integrated with automated laboratory platforms. In such scenario, digital decision errors translate directly into physical consequences. A single erroneous command affecting reagent sequencing, temperature control, or hazardous material handling can trigger chemical spills, fires, explosions, or personal injury [14]. This autonomy risk fundamentally distinguishes molecular discovery agents from general-purpose large language models. Their capacity for direct material manipulation demands rigorous risk control mechanisms, sustained human oversight, and a safety evaluation that is specifically tailored to autonomous molecule discovery.

For the systems surveyed here, we recommend a governance contract that records the verified molecular identity, allowed tools and resources, approval gates, stopping and abort conditions, and an auditable trace of inputs, intermediate states, tool versions, outputs, and overrides. This follows the lifecycle view of the NIST AI Risk Management Framework, whose core functions are to govern, map, measure, and manage risk [132]. Chemistry-specific implementations include safety tools in ChemCrow [91], governed skills in Mozi [14], and bounded laboratory interfaces in Coscientist [10]. Evaluation metrics should then capture not only task success but also invalid or unsafe episodes, blocked or escalated actions, budget violations, recovery events, and human interventions. These are proposed reporting requirements for this survey, not claims that any single control stack is sufficient for every laboratory.

5.2 Challenges in Molecular Perception

Molecular perception must preserve chemical identity while translating among representations selected for different tasks. This requirement becomes harder as agents move from human-checked L1 interactions to long, autonomous, and multimodal trajectories at L2 and above.

Representation fidelity across conversions. No single substrate exposes every relevant molecular property. SMILES and SELFIES are convenient for generation, graphs expose topology, 3D structures support physical reasoning, and images or spectra connect the agent to literature and experiments. Routing among these substrates can silently alter stereochemistry, protonation, tautomeric state, atom mapping, conformers, or measurement context. Structured formats such as MolJSON and MoleCode make connectivity more explicit [117, 157], but future agents still need round-trip validation, invariant checks, and persistent identity tracking at every representation boundary.

Adaptive representation and token granularity. Representation choice should depend on the next scientific action rather than on a fixed input format. A compact string may be sufficient for database lookup, whereas exact editing, docking, or spectroscopy may require graphs, coordinates, or multimodal state. Tokenization further determines whether edits correspond to characters, atoms, fragments, or structure-aware units. The open problem is to learn when the current representation is inadequate and to switch substrates without losing information, exceeding the context budget, or obscuring the action from human review.

Uncertainty-aware feedback perception. Tool outputs and experimental observations are not self-interpreting facts. Docking scores, property predictions, spectra, and assay measurements depend on model assumptions, units, conditions, and domains of validity. An

agent must retain these qualifications when converting an observation into its next state. Otherwise, an ambiguous experimental signal or an out-of-domain prediction can be collapsed into false certainty and propagated through later decisions. Perception benchmarks should therefore test identity preservation, unit and condition tracking, uncertainty retention, and recovery from conflicting multimodal evidence together with structure parsing accuracy.

5.3 Challenges in the Agent Framework

Although molecular agent design has progressed from tool-using dialogue to workflow control, reliable autonomous discovery remains unresolved. Open problems concern verifiability, memory, orchestration, evaluation, and safety.

Verifiable and molecule-native reasoning. Current agents can generate structured tool calls, executable code, protocols, and action traces, but they do not yet provide machine-checked chemical proofs or formal constraint certificates [69]. ChemActor represents synthesis procedures as structured chemical actions [177], but formal verification of molecule edits, synthesis constraints, safety rules, and protocol preconditions remains open. The reasoning substrate is also unresolved. Text and SMILES are interpretable but lossy, while graph, 3D, and latent states better preserve molecular structure but are harder to audit. MolLingo [98] and LatentChem [164] illustrate this tension between molecule-native expressiveness and interpretability.

Persistent memory and reliable reflection. Most memory remains local to one task, session, or campaign. ChemAgent [133] uses self-updating memories, TRACE [75] stores tool trajectories, and Augmented Memory [38] uses experience replay. L4 discovery would require cross-task scientific memory for failures, provenance, uncertainty, design rules, and reusable skills. Reflection faces a similar limitation. Systems such as ChatDrug [84], AgentDrug [61], Probe-Before-You-Edit [160], and MT-Mol [54] rely on external feedback, but high-autonomy agents often face delayed, noisy, or missing ground truth. Future evaluation of reflection must therefore consider calibration, stopping rules, and when the agent should defer to simulation, experiment, or human review.

Adaptive orchestration and evaluation. More tools, memory, and agents are not always better. Tooling-or-Not-Tooling shows that tool use can help some chemistry tasks while hurting others [166]. Orchestration should be adaptive: the controller must decide when to answer directly, retrieve evidence, call tools, coordinate specialists, or request human approval. Evaluation is also underdeveloped. Most benchmarks measure final task scores rather than controller behavior, such as reasoning traces, tool routing, planning under budget, memory reuse, reflection after failure, and multi-agent coordination. ChemLabs [156] and Tooling-or-Not-Tooling [166] are useful steps, but environment-grounded molecular-agent benchmarks remain limited.

Safety and governance. The controller-level problem is to enforce the boundaries defined above at the point of decision making. A final text filter cannot validate molecular identity, instrument state, or whether a requested action lies inside the approved domain. Dual-use molecular generation [143] and the expanding scope of self-driving laboratories [63] motivate structure-aware screening, permissioned execution, traceable decisions, and escalation when evidence or authority is insufficient. The open research question is how to assess these controls without relying solely on refusal rate as the safety metric.

5.4 Challenges in Molecular Toolboxes

Tools ground molecular agents in external evidence and executable actions, but they also define the operational boundary within which an agent can fail. A reliable toolbox must make capabilities, assumptions, costs, and hazards visible to the controller.

Semantic interoperability and state passing. Chemistry tools use heterogeneous identifiers, file formats, units, parameter conventions, and software environments. Ad hoc wrappers hide these differences but rarely guarantee that a state produced by one tool is valid input to the next. Shared, typed interfaces are needed for molecular identity, experimental conditions, uncertainty, provenance, and pre- and postconditions. General scientific tool infrastructures such as ToolUniverse [32] provide a useful foundation, but molecular workflows additionally require chemistry-aware schemas and validation across databases, simulators, synthesis planners, instruments, and robots.

Reliability, calibration, and failure recovery. Tool access does not guarantee trustworthy grounding. Databases can disagree, predictors can be out of domain, simulations can fail to converge, and laboratory interfaces can return incomplete observations. Agents must detect invalid inputs, timeouts, numerical failures, inconsistent outputs, and low confidence before using a result to revise the plan. Because tool augmentation can either help or hurt depending on the task [166], evaluation should measure tool selection, input construction, output interpretation, fallback behavior, and calibration in addition to final-answer accuracy.

Reproducibility, cost, and governed execution. Long molecular workflows require versioned tools, recorded parameters, environment metadata, and traceable state transitions so that a result can be reproduced rather than merely narrated. The controller must also trade off information value against latency, compute, assay cost, and instrument access. For synthesis or laboratory action, these resource policies must connect to the run-level permissions, abort conditions, and audit record defined in the safety governance subsection. The toolbox-specific challenge is implementing those controls across heterogeneous software and hardware interfaces while keeping execution interruptible and confined to the approved operational domain.

5.5 Challenges in Molecular Optimization

Molecular optimization tests whether an LLM-based molecular agent can improve candidates under scientific constraints, beyond generating valid structures. Agents must search a vast chemical space under noisy feedback, incomplete constraints, and conflicting objectives. Open problems lie in reward design, multi-objective search, action control, and workflow-level feedback.

Proxy objectives and reward hacking. Most optimization systems rely on surrogate objectives, such as QED, penalized LogP, docking

scores, predicted binding affinity, synthetic accessibility, and ADMET predictors. Early reinforcement-learning methods showed that molecular generators can be guided by property rewards [103, 111], while REINVENT4 supports richer multi-component scoring functions [88]. These scores provide fast feedback, but they remain proxies for the real scientific goal. An agent may exploit the evaluator and produce molecules that score well but are unstable, toxic, difficult to synthesize, or outside the predictor's reliable domain. The challenge is to design rewards that consider validity, synthesizability, uncertainty, diversity, and experimental plausibility rather than optimize a single scalar score.

Multi-objective trade-offs and local optima. Real molecular design usually involves conflicting objectives, where improving one property may weaken another. For example, higher potency may increase toxicity, while higher novelty may reduce synthesizability. DrugEx v2 [87], QADD [180], and MARS [155] reflect the move from single-objective optimization toward Pareto ranking, iterative multi-objective search, and graph-based editing. For agent-based optimization, premature exploitation is another concern. Once an agent finds a high-scoring scaffold, it may keep making small local edits around it rather than explore new chemical regions. Exploration-oriented methods such as Mol-AIR [159] and Augmented Hill-Climb [136] address part of this problem through novelty, coverage, or more efficient search. Future agents need better mechanisms for Pareto-aware optimization, diversity preservation, and control of exploration and exploitation.

Search efficiency and action-space control. Chemical space is too large for blind trial and error, especially when evaluation requires docking, molecular dynamics, DFT, retrosynthesis analysis, or laboratory validation. SMILES-based optimization is easy to combine with sequence models [88, 103, 111], but small token-level edits can invalidate a molecule or change it unexpectedly. Graph-based methods such as GCPN [165] and MolDQN [181] make actions more chemically explicit by operating on atoms, bonds, and graph edits. However, they also introduce larger and more constrained action spaces. The challenge is to define actions that are expressive enough for discovery, constrained enough to avoid invalid chemistry, and efficient enough for multi-round optimization.

Workflow-level feedback and credit assignment. Molecular optimization increasingly operates as a multi-step workflow rather than a single generation step. LLM-guided and tool-using systems such as ReMol [150], ChemCrow [91], ChemCRAFT [64], and MolClaw [174] show this shift from molecular generators toward broader agentic pipelines. When optimization fails, the cause may lie in generation, tool selection, reward modeling, planning, or molecular format conversion. This creates a workflow-level credit assignment problem. Evaluation should therefore measure final molecular scores together with multi-round improvement, tool-use efficiency, failure recovery, provenance, and robustness to unreliable feedback.

6 Future Directions

The open problems above become more useful when expressed as testable milestones rather than as another list of missing capabilities. We propose a progression in which each stage produces artifacts and measurements needed to justify the next expansion of authority.

Milestone 1: a verifiable molecular-state contract. Near-term systems should publish typed schemas for molecular identity, structure, conditions, units, uncertainty, provenance, and permitted edits, together with round-trip conversion tests across strings, graphs, coordinates, images, spectra, and experimental records. MolJSON, MoleCode, and MolLingo [98, 117, 157] illustrate current attempts to make molecular structure more explicit or molecule-native for LLM reasoning. The milestone is reached when declared conversions preserve identity and required metadata on a public test suite, invalid states are rejected before tool execution, and every downstream result can be traced to its input representation. This is a concrete prerequisite for attributing a later failure to the controller rather than to silent state corruption.

Milestone 2: reproducible and governed L2 loops. The next target is not a larger tool catalogue but a computational controller whose trajectories can be replayed and compared under the same evidence and resource budget. It should record tool versions, intermediate states, uncertainty, stopping decisions, and the reason for each escalation. Persistent memory can then be evaluated by withholding prior trajectories and measuring whether validated experience improves a new run without propagating stale or incorrect evidence. MolViBench and SciAgentBench expose multi-step tool execution [70, 120], while MADE supplies a budgeted closed-loop discovery setting [92]. A credible L2 milestone should report loop completion, improvement over rounds, recovery, cost, reproducibility, and mandatory human gates against fixed-workflow and no-memory baselines.

Milestone 3: bounded, safety-evaluated physical episodes. An L2 controller should progress to L3 only on a declared experimental domain with validated protocols, instrument limits, permissions, abort conditions, and measurement-quality checks. Coscientist and ORGANA provide primary examples of LLM-centered planning connected to physical execution and feedback [10, 22]; safety work on self-driving laboratories motivates evaluating the whole software–hardware–material system [63]. The milestone is repeatable completion of the same bounded episode across multiple trials while reporting execution success, measurement validity, recovery, human intervention, unsafe or invalid episodes, and resource use. A successful demonstration should also state what remains outside the operational domain.

Milestone 4: reliable iterative L3 campaigns. The stronger experimental target is a loop in which a measurement selects or changes a subsequent physical experiment. LLM-RDF and robotic ChemAgents demonstrate this pattern using measured yield or catalyst performance [116, 126]. Future work should compare the agent with fixed designs and established acquisition policies under matched budgets, replicate selected measurements, and report calibration, sample efficiency, stopping behavior, and recovery from failed or ambiguous readouts. This milestone separates an isolated automation success from an experimental policy that learns reliably during a campaign.

Milestone 5: evidence for cross-campaign scientific autonomy. L4 should remain an evidence standard rather than an aspirational label. A candidate system would need to preserve evidence across completed campaigns, formulate and prioritize a new question, choose among different computational and experimental task families, allocate resources, and revise the research agenda when results contradict its assumptions. Evaluation would require multiple task families, prospective rather than retrospective trials, independent scientific review, and evidence that the selected question and result are not recoverable from a fixed predefined workflow. Until such studies exist, progress is better described through the measurable L2 and L3 milestones above than through broad claims of autonomous discovery.

7 Conclusion

This survey developed a conceptual framework for molecular LLM agents from two complementary perspectives. The architectural view connects molecular representation and perception, an LLM-centered agent framework, domain-specific toolboxes, and learning and optimization. The scientific-autonomy view classifies agents by the outermost feedback loop they can reliably close without mandatory human intervention, from L1 assistive or fixed workflows to L4 scientific-agenda agents. Together, these views connect system design with demonstrated decision authority across computational and physical molecular workflows.

Our analysis indicates that planning, tool use, stronger language models, or multi-agent coordination alone do not establish scientific autonomy. Progress also requires chemically faithful perception, grounded and verifiable tool use, feedback-aware decision making, persistent provenance, realistic optimization, and safety within an explicit operational domain. Future evaluation should therefore consider task success together with loop completion, uncertainty, cost, reproducibility, human involvement, and safety, helping molecular agents advance from useful assistants toward trustworthy partners in molecular discovery.

References

[1] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C Smith, Berk Hess, and Erik Lindahl. 2015. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* 1 (2015), 19–25.

[2] Nawaf Alampara, Martiño Ríos-García, Chandan Gupta, Sajid Mannan, Santiago Miret, N. M. Anoop Krishnan, and Kevin Maik Jablonka. 2025. Task Alignment Outweighs Framework Choice in Scientific LLM Agents. In *NeurIPS 2025 Workshop on AI for Accelerated Materials Design*. https://openreview.net/forum?id=7cbwuA5k0T

[3] Mehrad Ansari and Seyed Mohamad Moosavi. 2024. Agent-based learning of materials datasets from the scientific literature. *Digital Discovery* 3, 12 (2024), 2607–2617.

[4] Mehrad Ansari, Jeffrey Watchorn, Carla E. Brown, and Joseph S. Brown. 2024. dZiner: Rational Inverse Design of Materials with AI Agents. arXiv:2410.03963 [physics.chem-ph] https://arxiv.org/abs/2410.03963

[5] Christoph Bannwarth, Eike Caldeweyher, Sebastian Ehlert, Andreas Hansen, Philipp Pracht, Jakob Seibert, Sebastian Spicher, and Stefan Grimme. 2021. Extended tight-binding quantum chemistry methods. *Wiley Interdisciplinary Reviews: Computational Molecular Science* 11, 2 (2021), e1493.

[6] Christoph Bannwarth, Sebastian Ehlert, and Stefan Grimme. 2019. GFN2-xTB—An accurate and broadly parametrized self-consistent tight-binding quantum chemical method with multipole electrostatics and density-dependent dispersion contributions. *Journal of chemical theory and computation* 15, 3 (2019), 1652–1671.

[7] Aaditya Baranwal, Akshaj Gupta, Yogesh S. Rawat, and Shruti Vyas. 2026. MolSight: Molecular property prediction with images. *arXiv preprint arXiv:2605.10157* (2026).

[8] A Patrícia Bento, Anne Hersey, Eloy Félix, Greg Landrum, Anna Gaulton, Francis Atkinson, Louisa J Bellis, Marleen De Veij, and Andrew R Leach. 2020. An open source chemical structure curation pipeline using RDKit. *Journal of cheminformatics* 12, 1 (2020), 51.

[9] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. 2000. The protein data bank. *Nucleic acids research* 28, 1 (2000), 235–242.

[10] Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. 2023. Autonomous chemical research with large language models. *Nature* 624, 7992 (2023), 570–578. doi:10.1038/s41586-023-06792-0

[11] Nathan Brown, Marwin Fiscato, Marwin H. S. Segler, and Alain C. Vaucher. 2019. GuacaMol: benchmarking models for de novo molecular design. *Journal of Chemical Information and Modeling* 59, 3 (2019), 1096–1108.

[12] Benjamin Burger, Phillip M. Maffettone, Vladimir V. Gusev, Catherine M. Aitchison, Yang Bai, Xiaoyan Wang, Xiaobo Li, Ben M. Alston, Buyi Li, Rob Clowes, Nicola Rankin, Brandon Harris, Reiner S. Sprick, and Andrew I. Cooper. 2020. A mobile robotic chemist. *Nature* 583 (2020), 237–241. doi:10.1038/s41586-020-2442-2

[13] Quintina Campbell, Sam Cox, Jorge Medina, Brittany Watterson, and Andrew D White. 2026. Mdcrow: Automating molecular dynamics workflows with large language models. *Machine Learning: Science and Technology* 7, 2 (2026), 025037. doi:10.1088/2632-2153/ae4b07

[14] He Cao, Siyu Liu, Fan Zhang, Zijing Liu, Hao Li, Bin Feng, Shengyuan Bai, Leqing Chen, Kai Xie, and Yu Li. 2026. Mozi: Governed autonomy for drug discovery llm agents. *arXiv preprint arXiv:2603.03655* (2026). https://arxiv.org/abs/2603.03655

[15] David A Case, Hasan Metin Aktulga, Kellon Belfon, David S Cerutti, G Andrés Cisneros, Vinícius Wilian D Cruzeiro, Negin Forouzesh, Timothy J Giese, Andreas W Gotz, Holger Gohlke, et al. 2023. AmberTools. *Journal of chemical information and modeling* 63, 20 (2023), 6183–6191.

[16] Kexin Chen, Jiamin Lu, Junyou Li, Xiaoran Yang, Yuyang Du, Kunyi Wang, Qiannuan Shi, Jiahui Yu, Lanqing Li, Jiezhong Qiu, et al. 2023. Chemist-X: Large language model-empowered agent for reaction condition recommendation in chemical synthesis. *arXiv preprint arXiv:2311.10776* (2023). doi:10.48550/arXiv.2311.10776

[17] Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. 2025. ScienceAgentBench: Toward Rigorous Assessment of Language Agents for Data-Driven Scientific Discovery. In *International Conference on Learning Representations*. https://proceedings.iclr.cc/paper_files/paper/2025/hash/f12b4df26344f3be803c06b555252efe-Abstract-Conference.html

[18] Austin H. Cheng, Andy Cai, Santiago Miret, Gustavo Malkomes, Mariano Phielipp, and Alán Aspuru-Guzik. 2023. Group SELFIES: a robust fragment-based molecular string representation. *Digital Discovery* 2, 3 (2023), 748–758.

[19] Yuan Chiang, Elvis Hsieh, Chia-Hong Chou, and Janosh Riebesell. 2025. LLaMP: Large Language Model Made Powerful for High-fidelity Materials Knowledge Retrieval. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 25189–25221. doi:10.18653/v1/2025.emnlp-main.1280

[20] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. 2020. ChemBERTa: Large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885* (2020).

[21] Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. 2022. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776* (2022).

[22] Kourosh Darvish, Marta Skreta, Yuchi Zhao, Naruki Yoshikawa, Sagnik Som, Miroslav Bogdanovic, Yang Cao, Han Hao, Haoping Xu, Alán Aspuru-Guzik, et al. 2025. ORGANA: A robotic assistant for automated chemistry experimentation and characterization. *Matter* 8, 2 (2025), 101897. doi:10.1016/j.matt.2024.10.015

[23] Vishal Dey, Xiao Hu, and Xia Ning. 2025. GeLLM^{3}O: Generalizing Large Language Models for Multi-property Molecule Optimization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, Vienna, Austria, 25192–25221. doi:10.18653/v1/2025.acl-long.1225

[24] Peter Eastman, Jason Swails, John D Chodera, Robert T McGibbon, Yutong Zhao, Kyle A Beauchamp, Lee-Ping Wang, Andrew C Simmonett, Matthew P Harrigan, Chaya D Stern, et al. 2017. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS computational biology* 13, 7 (2017), e1005659.

[25] Carl Edwards, Chi Han, Gawon Lee, Thao Nguyen, Bowen Jin, Chetan Kumar Prasad, Sara Szymkuć, Bartosz A Grzybowski, Ying Diao, Jiawei Han, et al. 2025. mclm: A function-infused and synthesis-friendly modular chemical language model. *arXiv e-prints* (2025), arXiv–2505.

[26] Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. 2022. Translation between molecules and natural language. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 375–413.

[27] Yao Fehlis, Charles Crain, Aidan Jensen, Michael Watson, James Juhasz, Paul Mandel, Betty Liu, Shawn Mahon, Daren Wilson, Nick Lynch-Jonely, et al. 2025. Accelerating drug discovery through agentic ai: A multi-agent approach to laboratory automation in the dmta cycle. *arXiv preprint arXiv:2507.09023* (2025).

[28] Yao Fehlis, Charles Crain, Aidan Jensen, Michael Watson, James Juhasz, Paul Mandel, Betty Liu, Shawn Mahon, Daren Wilson, Nick Lynch-Jonely, et al. 2025. Technical Implementation of Tippy: Multi-Agent Architecture and System Design for Drug Discovery Laboratory Automation. *arXiv preprint arXiv:2507.17852* (2025).

[29] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox. 2016. Gaussian~16 Revision C.01. Gaussian Inc. Wallingford CT.

[30] Bowen Gao, Yanwen Huang, Yiqiao Liu, Wenxuan Xie, Wei-Ying Ma, Ya-Qin Zhang, and Yanyan Lan. 2025. Pharmagents: Building a virtual pharma with large language model agents. *arXiv preprint arXiv:2503.22164* (2025).

[31] Bowen Gao, Yanwen Huang, Yiqiao Liu, Wenxuan Xie, Wei-Ying Ma, Ya-Qin Zhang, and Yanyan Lan. 2025. Pushing the boundaries of structure-based drug design through collaboration with large language models. *arXiv preprint arXiv:2503.01376* (2025).

[32] Shanghua Gao, Richard Zhu, Pengwei Sui, Zhenglun Kong, Sufian Aldogom, Yepeng Huang, Ayush Noori, Reza Shamji, Krishna Parvataneni, Theodoros Tsiligkaridis, et al. 2025. Democratizing AI scientists using ToolUniverse. *arXiv preprint arXiv:2509.23426* (2025).

[33] Samuel Genheden, Amol Thakkar, Veronika Chadimová, Jean-Louis Reymond, Ola Engkvist, and Esben Bjerrum. 2020. AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning. *Journal of cheminformatics* 12, 1 (2020), 70.

[34] Rafael Gómez-Bombarelli, Jennifer N. Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alán Aspuru-Guzik. 2018. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science* 4, 2 (2018), 268–276. doi:10.1021/acscentsci.7b00572

[35] Vitor F Grizzi, Thang Duc Pham, Luke N Pretzie, Jiayi Xu, Murat Keceli, and Cong Liu. 2026. ChemGraph-XANES: An Agentic Framework for XANES Simulation and Analysis. *arXiv preprint arXiv:2604.16205* (2026).

[36] Sergey A Guda, Alexander A Guda, Mikhail A Soldatov, Kirill A Lomachenko, Aram L Bugaev, Carlo Lamberti, Wojciech Gawelda, Christian Bressler, Grigory Smolentsev, Alexander V Soldatov, et al. 2015. Optimized finite difference method for the full-potential XANES simulations: Application to molecular adsorption geometries in MOFs and metal–ligand intersystem crossing transients. *Journal of chemical theory and computation* 11, 9 (2015), 4512–4521.

[37] Salomé Guilbert, Cassandra Masschelein, Jeremy Goumaz, Bohdan Naida, and Philippe Schwaller. 2025. DynaMate: An Autonomous Agent for Protein-Ligand Molecular Dynamics Simulations. *arXiv preprint arXiv:2512.10034* (2025).

[38] Jeff Guo and Philippe Schwaller. 2024. Augmented memory: sample-efficient generative molecular design with reinforcement learning. *Jacs Au* 4, 6 (2024), 2160–2172.

[39] Shuhan Guo, Yatao Bian, Ruibing Wang, Nan Yin, Zhen Wang, and Quanming Yao. 2024. UniMoT: Unified molecule-text language model with discrete token representation. *arXiv preprint arXiv:2408.00863* (2024).

[40] Florian Häse, Loïc M. Roch, and Alán Aspuru-Guzik. 2018. Phoenics: A Bayesian optimizer for chemistry. *ACS Central Science* 4, 9 (2018), 1134–1145. doi:10.1021/acscentsci.8b00307

[41] Jiyan He, Weitao Feng, Yaosen Min, Jingwei Yi, Kunsheng Tang, Shuai Li, Jie Zhang, Kejiang Chen, Wenbo Zhou, Xing Xie, Weiming Zhang, Nenghai Yu, and Shuxin Zheng. 2023. Control Risk for Potential Misuse of Artificial Intelligence in Science. arXiv:2312.06632 [cs.AI] https://arxiv.org/abs/2312.06632

[42] Stephen R. Heller, Alan McNaught, Igor Pletnev, Stephen Stein, and Dmitrii Tchekhovskoi. 2015. InChI, the IUPAC International Chemical Identifier. *Journal of Cheminformatics* 7, 1 (2015), 23.

[43] Ask Hjorth Larsen, Jens Jørgen Mortensen, Jakob Blomqvist, Ivano E Castelli, Rune Christensen, Marcin Dułak, Jesper Friis, Michael N Groves, Bjørk Hammer, Cory Hargus, et al. 2017. The atomic simulation environment—a Python library for working with atoms. *Journal of Physics: Condensed Matter* 29, 27 (2017), 273002.

[44] Hisayuki Horai, Masanori Arita, Shigehiko Kanaya, Yoshito Nihei, Tasuku Ikeda, Kazuhiro Suwa, Yuya Ojima, Kenichi Tanaka, Satoshi Tanaka, Ken Aoshima, et al. 2010. MassBank: a public repository for sharing mass spectral data for life sciences. *Journal of mass spectrometry* 45, 7 (2010), 703–714.

[45] Xiuyuan Hu, Guoqing Liu, Yang Zhao, and Hao Zhang. 2023. De novo drug design using reinforcement learning with multiple gpt agents. *Advances in Neural Information Processing Systems* 36 (2023), 7405–7418.

[46] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W. Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. 2021. Therapeutics Data Commons: machine learning datasets and tasks for drug discovery and development. *arXiv preprint arXiv:2102.09548* (2021).

[47] Linda Hung, Joyce A. Yager, Danielle Monteverde, Dave Baiocchi, Ha-Kyung Kwon, Shijing Sun, and Santosh Suram. 2024. Autonomous laboratories for accelerated materials discovery: A community survey and practical insights. *Digital Discovery* 3 (2024), 1273–1279. doi:10.1039/D4DD00059E

[48] Yoshitaka Inoue, Tianci Song, Xinling Wang, Augustin Luna, and Tianfan Fu. 2025. Drugagent: Multi-agent large language model-based reasoning for drug-target interaction prediction. *ArXiv* (2025), arXiv–2408.

[49] John J Irwin, Khanh G Tang, Jennifer Young, Chinzorig Dandarchuluun, Benjamin R Wong, Munkhzul Khurelbaatar, Yurii S Moroz, John Mayfield, and Roger A Sayle. 2020. ZINC20—a free ultralarge-scale chemical database for ligand discovery. *Journal of chemical information and modeling* 60, 12 (2020), 6065–6073.

[50] Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, et al. 2013. Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. *APL materials* 1, 1 (2013).

[51] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2018. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*. PMLR, 2323–2332.

[52] Y Joly. 2001. X-ray absorption near-edge structure calculations beyond the muffin-tin approximation. *Physical Review B* 63, 12 (2001), 125120.

[53] Yeonghun Kang and Jihan Kim. 2024. ChatMOF: an artificial intelligence system for predicting and generating metal-organic frameworks using large language models. *Nature communications* 15, 1 (2024), 4705.

[54] Hyomin Kim, Yunhui Jang, and Sungsoo Ahn. 2025. Mt-mol: Multi agent system with tool-based reasoning for molecular optimization. *Artificial Intelligence Repository* (2025).

[55] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. 2023. PubChem 2023 update. *Nucleic acids research* 51, D1 (2023), D1373–D1380.

[56] Takayuki Kimura. 2026. VQ-Atom: Semantic discretization of local atomic environments for molecular representation learning. *arXiv preprint arXiv:2605.16823* (2026).

[57] Stephen T. Knox, Sam J. Parkinson, Clarissa Y. P. Wilding, Richard A. Bourne, and Nicholas J. Warren. 2022. Autonomous polymer synthesis delivered by multi-objective closed-loop optimisation. *Polymer Chemistry* 13, 11 (2022), 1576–1585. doi:10.1039/D2PY00040G

[58] Mario Krenn, Florian Hase, AkshatKumar Nigam, Pascal Friederich, and Alán Aspuru-Guzik. 2020. SELFIES: a robust representation of semantically constrained graphs with an example application in chemistry. *Machine Learning: Science and Technology* 1, 4 (2020), 045024.

[59] Stefan Kuhn, Heinz Kolshorn, Christoph Steinbeck, and Nils Schlörer. 2024. Twenty years of nmrshiftdb2: A case study of an open database for analytical chemistry. *Magnetic Resonance in Chemistry* 62, 2 (2024), 74–83.

[60] Vinay Kumar, Satyendra Rajput, Mausam, and N. M. Anoop Krishnan. 2026. MDGym: Benchmarking AI Agents on Molecular Simulations. *arXiv preprint arXiv:2605.08941* (2026). arXiv:2605.08941 [cs.AI] https://arxiv.org/abs/2605.08941

[61] Khiem Le, Ting Hua, and Nitesh V Chawla. 2024. AgentDrug: Utilizing Large Language Models in An Agentic Workflow for Zero-Shot Molecular Optimization. *arXiv preprint arXiv:2410.13147* (2024).

[62] Namkyeong Lee, Edward De Brouwer, Ehsan Hajiramezanali, Tommaso Biancalani, Chanyoung Park, and Gabriele Scalia. 2026. Rag-enhanced collaborative llm agents for drug discovery. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 40. 561–569.

[63] Shi Xuan Leong, Caleb E. Griesbach, Rui Zhang, Kourosh Darvish, Yuchi Zhao, Abhijoy Mandal, Yunheng Zou, Han Hao, Varinia Bernales, Alán Aspuru-Guzik, et al. 2025. Steering towards safe self-driving laboratories. *Nature Reviews Chemistry* 9 (2025), 707–722. doi:10.1038/s41570-025-00747-x

[64] Hao Li, He Cao, Shenyao Peng, Zijing Liu, Bin Feng, Yu Wang, Zhiyuan Yan, Yonghong Tian, Yu Li, and Li Yuan. 2026. Agentic reinforcement learning empowers next-generation chemical language models for molecular design and synthesis. *arXiv preprint arXiv:2601.17687* (2026).

[65] Jiatong Li, Junxian Li, Weida Wang, Yunqing Liu, Changmeng Zheng, Yatao Bian, Dongzhan Zhou, Xiao-Yong Wei, and Qing Li. 2026. Speak-to-structure: Evaluating llms in open-domain natural language-driven molecule generation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 9314–9325.

[66] Jiatong Li, Wei Liu, Zhihao Ding, Wenqi Fan, Yuqiang Li, and Qing Li. 2025. Large language models are in-context molecule learners. *IEEE Transactions on Knowledge and Data Engineering* (2025).

[67] Jiatong Li, Yunqing Liu, Wenqi Fan, Xiao-Yong Wei, Hui Liu, Jiliang Tang, and Qing Li. 2024. Empowering molecule discovery for molecule-caption translation with large language models: A chatgpt perspective. *IEEE transactions on knowledge and data engineering* 36, 11 (2024), 6071–6083.

[68] Jiatong Li, Yunqing Liu, Wei Liu, Jingdi Lei, Di Zhang, Wenqi Fan, Dongzhan Zhou, Yuqiang Li, and Qing Li. 2026. Molreflect: Towards in-context fine-grained alignments between molecules and texts. *IEEE Transactions on Knowledge and Data Engineering* (2026).

[69] Jiatong Li, Yuxuan Ren, Weida Wang, Xiaoyong Wei, and Yatao Bian. 2026. Chemical Chain-of-Thought Functions as a Hallucination-Prone Molecular Scratchpad. *arXiv preprint arXiv:2607.20935* (2026).

[70] Jiatong Li, Yuxuan Ren, Weida Wang, Changmeng Zheng, Xiao-yong Wei, Qing Li, and Yatao Bian. 2026. MolViBench: Evaluating LLMs on Molecular Vibe Coding. *arXiv preprint arXiv:2605.02351* (2026).

[71] Jiatong Li, Weida Wang, Qinggang Zhang, Junxian Li, Di Zhang, Changmeng Zheng, Shufei Zhang, Xiaoyong Wei, and Qing Li. 2025. Mol-r1: Towards explicit long-cot reasoning in molecule discovery. *arXiv preprint arXiv:2508.08401* (2025).

[72] Jiatong Li, Weida Wang, Changmeng Zheng, Shufei Zhang, Yatao Bian, Xiao yong Wei, and Qing Li. 2026. Do LLMs Truly Generalize in the Molecular Domain? A Perturbation-Based Analysis. *arXiv preprint arXiv:2607.01800* (2026).

[73] Junxian Li, Di Zhang, Xunzhi Wang, Zeying Hao, Jingdi Lei, Qian Tan, Cai Zhou, Wei Liu, Yaotian Yang, Xinrui Xiong, Weiyun Wang, Zhe Chen, Wenhai Wang, Wei Li, Shufei Zhang, Mao Su, Wanli Ouyang, Yuqiang Li, and Dongzhan Zhou. 2025. ChemVLM: Exploring the Power of Multimodal Large Language Models in Chemistry Area. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 415–423. doi:10.1609/aaai.v39i1.32020

[74] Kun Li, Zhennan Wu, Shoupeng Wang, Jia Wu, Shirui Pan, and Wenbin Hu. 2025. DrugPilot: LLM-based parameterized reasoning agent for drug discovery. *arXiv preprint arXiv:2505.13940* (2025).

[75] Lingxiao Li, Haobo Zhang, Ruohao Fan, Bin Chen, and Jiayu Zhou. 2026. Molecular Lead Optimization via Agentic Tool Planning. *arXiv preprint arXiv:2605.28862* (2026).

[76] Yizhan Li, Florence Cloutier, Sifan Wu, Ali Parviz, Boris Knyazev, Yan Zhang, Glen Berseth, and Bang Liu. 2026. Mˆ 4olGen: Multi-Agent, Multi-Stage Molecular Generation under Precise Multi-Property Constraints. *arXiv preprint arXiv:2601.10131* (2026).

[77] Zhucong Li, Jin Xiao, Bowei Zhang, Zhijian Zhou, Qianyu He, Fenglei Cao, Jiaqing Liang, and Yuan Qi. 2025. Chemhts: Hierarchical tool stacking for enhancing chemical agents. *arXiv preprint arXiv:2502.14327* (2025).

[78] Zhucong Li, Bowei Zhang, Jin Xiao, Zhijian Zhou, Fenglei Cao, Jiaqing Liang, and Yuan Qi. 2025. Chemhas: Hierarchical agent stacking for enhancing chemistry tools. *arXiv preprint arXiv:2505.21569v1* (2025).

[79] P Linstorm. 1998. NIST chemistry webbook, NIST standard reference database number 69. *J. Phys. Chem. Ref. Data, Monograph* 9 (1998), 1–1951.

[80] Gang Liu, Michael Sun, Wojciech Matusik, Meng Jiang, and Jie Chen. 2025. Multimodal large language models for inverse molecular design with retrosynthetic planning. In *International Conference on Learning Representations*, Vol. 2025. 41744–41771.

[81] Siyu Liu, Bo Hu, Beilin Ye, Jiamin Xu, David J Srolovitz, and Tongqi Wen. 2025. Mattools: Benchmarking large language models for materials science tools. *arXiv preprint arXiv:2505.10852* (2025).

[82] Sizhe Liu, Yizhou Lu, Siyu Chen, Xiyang Hu, Jieyu Zhao, Yingzhou Lu, and Yue Zhao. 2024. Drugagent: Automating ai-aided drug discovery programming through llm multi-agent collaboration. *arXiv preprint arXiv:2411.15692* (2024).

[83] Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. 2022. Pre-training molecular graph representation with 3D geometry. In *International Conference on Learning Representations*.

[84] Shengchao Liu, Jiongxiao Wang, Yijin Yang, Chengpeng Wang, Ling Liu, Hongyu Guo, and Chaowei Xiao. 2024. Conversational Drug Editing Using Retrieval and Domain Feedback. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=yRrPfKyJQ2

[85] Tianyu Liu, Allen Xin Wang, Antonia Panescu, Lisa Xinyi Chen, Wenxin Long, Xinyu Wei, et al. 2026. Benchmarking AI Agents for Addressing Scientific Challenges Across Scales. *arXiv preprint arXiv:2606.12736* (2026). https://arxiv.org/abs/2606.12736

[86] Wei Liu, Jiangtao Feng, Hongli Yu, Yuxuan Song, Yuqiang Li, Shufei Zhang, Lei Bai, Wei-Ying Ma, and Hao Zhou. 2026. Retro-r1: LLM-based agentic retrosynthesis. *Advances in Neural Information Processing Systems* 38 (2026), 70709–70737.

[87] Xuhan Liu, Kexin Ye, Herman W. T. van Vlijmen, Adriaan P. IJzerman, and Gerard J. P. van Westen. 2021. DrugEx v2: De novo design of drug molecules by Pareto-based multi-objective reinforcement learning in polypharmacology. *Journal of Cheminformatics* 13 (2021), 85. doi:10.1186/s13321-021-00561-9

[88] Hannes H. Loeffler, Jiazhen He, Alessandro Tibo, Jon Paul Janet, Artur Voronov, Lewis H. Mervin, Ola Engkvist, and Hongming Chen. 2024. REINVENT4: Modern AI-driven generative molecule design. *Journal of Cheminformatics* 16 (2024), 20. doi:10.1186/s13321-024-00812-5

[89] Daniel M Lowe, Peter T Corbett, Peter Murray-Rust, and Robert C Glen. 2011. Chemical name to structure: OPSIN, an open source solution.

[90] Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K. Qiu, and Yuqing Yang. 2025. Agent Lightning: Train ANY AI Agents with Reinforcement Learning. *arXiv preprint arXiv:2508.03680* (2025).

[91] Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. 2024. Augmenting large language models with chemistry tools. *Nature machine intelligence* 6, 5 (2024), 525–535. doi:10.1038/s42256-024-00832-8

[92] Shreshth A. Malik, Tiarnan Doherty, Panagiotis Tigas, Muhammed Razzak, Stephen J. Roberts, Aron Walsh, and Yarin Gal. 2026. MADE: Benchmark Environments for Closed-Loop Materials Discovery. In *Proceedings of the 43rd International Conference on Machine Learning*, Vol. 306. https://arxiv.org/abs/2601.20996

[93] Andrew D McNaughton, Gautham Krishna Sankar Ramalaxmi, Agustin Kruel, Carter R Knutson, Rohith A Varikoti, and Neeraj Kumar. 2024. Cactus: Chemistry agent connecting tool usage to science. *ACS omega* 9, 46 (2024), 46563–46573. doi:10.1021/acsomega.4c08408

[94] Andrew T McNutt, Paul Francoeur, Rishal Aggarwal, Tomohide Masuda, Rocco Meli, Matthew Ragoza, Jocelyn Sunseri, and David Ryan Koes. 2021. GNINA 1.0: molecular docking with deep learning. *Journal of cheminformatics* 13, 1 (2021), 43.

[95] Siddharth Narayanan, James Braza, Ryan-Rhys Griffiths, Albert Bou, Geemi Wellawatte, Mayk Caldas Ramos, Ludovico Mitchener, Michael Pieler, Sam Rodriques, and Andrew White. 2026. Training a scientific reasoning model for chemistry. *Advances in Neural Information Processing Systems* 38 (2026), 157671–157710.

[96] Frank Neese, Frank Wennmohs, Ute Becker, and Christoph Riplinger. 2020. The ORCA quantum chemistry program package. *The Journal of chemical physics* 152, 22 (2020).

[97] Steffen Neumann, René Meier, Michael Wenk, Anjana Elapavalore, Takaaki Nishioka, Tobias Schulze, Michael Stravs, Hiroshi Tsugawa, Fumio Matsuda, and Emma L Schymanski. 2026. MassBank: an open and FAIR mass spectral data resource. *Nucleic Acids Research* 54, D1 (2026), D601–D606.

[98] Thao Nguyen and Heng Ji. 2026. MolLingo: Molecule-Native Representations for LLM-Powered Scientific Agents. *arXiv preprint arXiv:2605.27853* (2026). https://arxiv.org/abs/2605.27853

[99] Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, et al. 2025. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 6140–6150.

[100] Emmanuel Noutahi, Cristian Gabellini, Michael Craig, Jonathan S. C. Lim, and Prudencio Tossou. 2023. Gotta be SAFE: a new framework for molecular design. *arXiv preprint arXiv:2310.10773* (2023).

[101] Noel M O'Boyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch, and Geoffrey R Hutchison. 2011. Open Babel: An open chemical toolbox. *Journal of cheminformatics* 3, 1 (2011), 33.

[102] Janghoon Ock, Radheesh Sharma Meda, Srivathsan Badrinarayanan, Neha S Aluru, Achuth Chandrasekhar, and Amir Barati Farimani. 2026. Large language model agent for modular task execution in drug discovery. *Journal of Chemical Information and Modeling* 66, 4 (2026), 2055–2068.

[103] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of Cheminformatics* 9, 1 (2017), 48. doi:10.1186/s13321-017-0235-x

[104] Shyue Ping Ong, William Davidson Richards, Anubhav Jain, Geoffroy Hautier, Michael Kocher, Shreyas Cholia, Dan Gunter, Vincent L Chevrier, Kristin A Persson, and Gerbrand Ceder. 2013. Python Materials Genomics (pymatgen): A robust, open-source python library for materials analysis. *Computational Materials Science* 68 (2013), 314–319.

[105] Qihua Pan, Dong Xu, Qianwei Yang, Jenna Xinyi Yao, Sisi Yuan, Zexuan Zhu, Jianqiang Li, and Junkai Ji. 2026. FROGENT: An End-to-End Full-process Drug Design Multi-Agent System. arXiv:2508.10760 [q-bio.BM] https://arxiv.org/abs/2508.10760

[106] Gihan Panapitiya, Emily Saldanha, Heather Job, and Olivia Hess. 2026. Autolabs: Cognitive multi-agent systems with self-correction for autonomous chemical experimentation. *Scientific Reports* 16, 1 (2026), 19554. doi:10.1038/s41598-026-45593-z

[107] Robert M Parrish, Lori A Burns, Daniel GA Smith, Andrew C Simmonett, A Eugene DePrince III, Edward G Hohenstein, Ugur Bozkaya, Alexander Yu Sokolov, Roberto Di Remigio, Ryan M Richard, et al. 2017. Psi4 1.1: An open-source electronic structure program emphasizing automation, advanced libraries, and interoperability. *Journal of chemical theory and computation* 13, 7 (2017), 3185–3197.

[108] Iman Peivaste, Ahmed Makradi, and Salim Belouettar. 2026. ChemNavigator: Agentic AI Discovery of Design Rules for Organic Photocatalysts. *arXiv preprint arXiv:2601.17084* (2026).

[109] Thang D Pham, Aditya Tanikanti, and Murat Keçeli. 2026. ChemGraph as an agentic framework for computational chemistry workflows. *Communications Chemistry* (2026).

[110] Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, et al. 2020. Molecular sets (MOSES): a benchmarking platform for molecular generation models. *Frontiers in Pharmacology* 11 (2020), 565644.

[111] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. 2018. Deep reinforcement learning for de novo drug design. *Science Advances* 4, 7 (2018), eaap7885. doi:10.1126/sciadv.aap7885

[112] Yujie Qian, Jiang Guo, Zhengkai Tu, Connor W. Coley, and Regina Barzilay. 2023. MolScribe: Robust molecular structure recognition with image-to-graph generation. In *International Conference on Learning Representations*.

[113] Nian Ran, Yue Wang, Xiaoyuan Zhang, Zhongzheng Li, Qingsong Ran, Wenhao Li, and Richard Allmendinger. 2025. ExLLM: Experience-Enhanced LLM Optimization for Molecular Design and Beyond. *arXiv preprint arXiv:2502.12845* (2025).

[114] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. 2022. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence* 4, 12 (2022), 1256–1264.

[115] Samuel Rothfarb, Megan C Davis, Ivana Matanovic, Baikun Li, Edward F Holby, and Wilton JM Kort-Kamp. 2025. Hierarchical Multi-agent Large Language Model Reasoning for Autonomous Functional Materials Discovery. *arXiv preprint arXiv:2512.13930* (2025).

[116] Yixiang Ruan, Chenyin Lu, Ning Xu, Yuchen He, Yixin Chen, Jian Zhang, Jun Xuan, Jianzhang Pan, Qun Fang, Hanyu Gao, et al. 2024. An automatic end-to-end chemical synthesis development platform powered by large language models. *Nature communications* 15, 1 (2024), 10160. doi:10.1038/s41467-024-54457-x

[117] Nicholas T. Runcie, Fergus Imrie, and Charlotte M. Deane. 2026. Molecular representations for large language models. *arXiv preprint arXiv:2605.01822* (2026). https://arxiv.org/abs/2605.01822

[118] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Advances in Neural Information Processing Systems*, Vol. 36. 68539–68551.

[119] Kristof T. Schütt, Pieter-Jan Kindermans, Huziel E. Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. 2017. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Advances in Neural Information Processing Systems*, Vol. 30.

[120] Yujiong Shen, Yajie Yang, Zhiheng Xi, Binze Hu, Huayu Sha, Jiazheng Zhang, Qiyuan Peng, Junlin Shang, Jixuan Huang, Yutao Fan, Jingqi Tong, Shihan Dou, Ming Zhang, Lei Bai, Zhenfei Yin, Tao Gui, Xingjun Ma, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. 2026. SciAgentGym: Benchmarking Multi-Step Scientific Tool-use in LLM Agents. *arXiv preprint arXiv:2602.12984* (2026). arXiv:2602.12984 [cs.CL] https://arxiv.org/abs/2602.12984

[121] Zhuofan Shi, Yufei Shao, Mengyan Dai, Yadong Yu, Pan Xiang, Dongliang Huang, Hongxu An, Chunxiao Xin, Haiyang Shen, Zhenyu Wang, et al. 2026. MDAgent2: Large Language Model for Code Generation and Knowledge Q&A in Molecular Dynamics. *arXiv preprint arXiv:2601.02075* (2026).

[122] Benjamin J. Shields, Jason Stevens, Jun Li, Marvin Parasram, Farhan Damani, Jesus I. M. Alvarado, Jacob M. Janey, Ryan P. Adams, and Abigail G. Doyle. 2021. Bayesian reaction optimization as a tool for chemical synthesis. *Nature* 590 (2021), 89–96. doi:10.1038/s41586-021-03213-y

[123] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366* (2023).

[124] Zachary S. Siegel, Sayash Kapoor, Nitya Nadgir, Benedikt Stroebl, and Arvind Narayanan. 2024. CORE-Bench: Fostering the Credibility of Published Research Through a Computational Reproducibility Agent Benchmark. *arXiv preprint arXiv:2409.11363* (2024). https://arxiv.org/abs/2409.11363

[125] Gleb V Solovev, Alina B Zhidkovskaya, Anastasia Orlova, Nina Gubina, Anastasia Vepreva, Rodion Golovinskii, Ilya Tonkii, Ivan Dubrovsky, Ivan Gurev, Dmitry Gilemkhanov, et al. 2025. MADD: Multi-Agent Drug Discovery Orchestra. In *Findings of the Association for Computational Linguistics: EMNLP 2025*. 6956–6998.

[126] Tao Song, Man Luo, Xiaolong Zhang, Linjiang Chen, Yan Huang, Jiaqi Cao, Qing Zhu, Daobin Liu, Baicheng Zhang, Gang Zou, et al. 2025. A multiagent-driven robotic AI chemist enabling autonomous chemical research on demand. *Journal of the American Chemical Society* 147, 15 (2025), 12534–12545. doi:10.1021/jacs.4c17738

[127] Henry Sprueill, Carl Edwards, Mariefel Olarte, Udishnu Sanyal, Heng Ji, and Sutanay Choudhury. 2023. Monte carlo thought search: Large language model querying for complex scientific reasoning in catalyst design. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 8348–8365.

[128] Henry W Sprueill, Carl Edwards, Khushbu Agarwal, Mariefel V Olarte, Udishnu Sanyal, Conrad Johnston, Hongbin Liu, Heng Ji, and Sutanay Choudhury. 2024. ChemReasoner: Heuristic search over a large language model's knowledge space using quantum-chemical feedback. *arXiv preprint arXiv:2402.10980* (2024).

[129] Christoph Steinbeck, Stefan Krause, and Stefan Kuhn. 2003. NMRShiftDB constructing a free chemical information system with open-source components. *Journal of chemical information and computer sciences* 43, 6 (2003), 1733–1739.

[130] Minyi Su, Qifan Yang, Yu Du, Guoqin Feng, Zhihai Liu, Yan Li, and Renxiao Wang. 2018. Comparative assessment of scoring functions: the CASF-2016 update. *Journal of chemical information and modeling* 59, 2 (2018), 895–913.

[131] Nathan J. Szymanski, B. Rendy, Y. Fei, R. E. Kumar, T. He, D. Milsted, M. J. McDermott, M. Gallant, E. D. Cubuk, A. Merchant, et al. 2023. An autonomous laboratory for the accelerated synthesis of novel materials. *Nature* 624 (2023), 86–91. doi:10.1038/s41586-023-06734-w

[132] Elham Tabassi. 2023. *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. Technical Report NIST AI 100-1. National Institute of Standards and Technology. doi:10.6028/NIST.AI.100-1

[133] Xiangru Tang, Tianyu Hu, Muyang Ye, Yanjun Shao, Xunjian Yin, Siru Ouyang, Wangchunshu Zhou, Pan Lu, Zhuosheng Zhang, Yilun Zhao, et al. 2025. Chemagent: Self-updating memories in large language models improves chemical reasoning. In *The Thirteenth International Conference on Learning Representations*.

[134] Xiangru Tang, Qiao Jin, Kunlun Zhu, Tongxin Yuan, Yichi Zhang, Wangchunshu Zhou, Meng Qu, Yilun Zhao, Jian Tang, Zhuosheng Zhang, Arman Cohan, Zhiyong Lu, and Mark B. Gerstein. 2024. Risks of AI scientists: prioritizing safeguarding over autonomy. *Nature Communications* 16 (2024). doi:10.1038/s41467-025-63913-1

[135] Asu Büşra Temizer, Gökçe Uludoğan, Rıza Özçelik, Taha Koulani, Elif Ozkirimli, Kutlu O. Ulgen, Nilgün Karalı, and Arzucan Özgür. 2022. Exploring data-driven chemical SMILES tokenization approaches to identify key protein-ligand binding moieties. *arXiv preprint arXiv:2210.14642* (2022).

[136] Morgan Thomas et al. 2022. Augmented Hill-Climb increases reinforcement learning efficiency for language-based de novo molecule generation. *Journal of Cheminformatics* 14 (2022), 68. doi:10.1186/s13321-022-00646-z

[137] Aidan P Thompson, H Metin Aktulga, Richard Berger, Dan S Bolintineanu, W Michael Brown, Paul S Crozier, Pieter J In't Veld, Axel Kohlmeyer, Stan G Moore, Trung Dac Nguyen, et al. 2022. LAMMPS-a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Computer physics communications* 271 (2022), 108171.

[138] Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. 2024. SciCode: A Research Coding Benchmark Curated by Scientists. In *Advances in Neural Information Processing Systems*, Vol. 37. doi:10.52202/079017-0963 Datasets and Benchmarks Track.

[139] Benjamin I Tingle, Khanh G Tang, Mar Castanon, John J Gutierrez, Munkhzul Khurelbaatar, Chinzorig Dandarchuluun, Yurii S Moroz, and John J Irwin. 2023. ZINC-22: A free multi-billion-scale database of tangible compounds for ligand discovery. *Journal of chemical information and modeling* 63, 4 (2023), 1166–1176.

[140] Oleg Trott and Arthur J Olson. 2010. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry* 31, 2 (2010), 455–461.

[141] Zhengkai Tu, Sourabh J Choure, Mun Hong Fong, Jihye Roh, Itai Levin, Kevin Yu, Joonyoung F Joung, Nathan Morgan, Shih-Cheng Li, Xiaoqi Sun, et al. 2025. ASKCOS: an open source software suite for synthesis planning. *arXiv preprint arXiv:2501.01835* (2025).

[142] UniProt Consortium. 2023. UniProt: the universal protein knowledgebase in 2023. *Nucleic acids research* 51, D1 (2023), D523–D531.

[143] Fabio Urbina, Filippa Lentzos, Cédric Invernizzi, and Sean Ekins. 2022. Dual use of artificial-intelligence-powered drug discovery. *Nature Machine Intelligence* 4 (2022), 189–191. doi:10.1038/s42256-022-00465-9

[144] Ivana Vichentijevikj, Kostadin Mishev, and Monika Simjanoska Misheva. 2026. Prompt-to-pill: Multi-Agent drug discovery and clinical simulation pipeline. *Bioinformatics Advances* 6, 1 (2026), vbaf323.

[145] Amanda A. Volk and Milad Abolhasani. 2024. Performance metrics to unleash the power of self-driving labs in chemistry and materials science. *Nature Communications* 15 (2024), 1378. doi:10.1038/s41467-024-45569-5

[146] Amanda A. Volk, Robert W. Epps, Daniel T. Yonemoto, Benjamin S. Masters, Felix Castellano, Kristofer G. Reyes, and Milad Abolhasani. 2023. AlphaFlow: Autonomous discovery and optimization of multi-step chemistry using a self-driven fluidic lab guided by reinforcement learning. *Nature Communications* 14, 1 (2023), 1403. doi:10.1038/s41467-023-37139-y

[147] Alexius Wadell, Anoushka Bhutani, and Venkatasubramanian Viswanathan. 2024. Tokenization for molecular foundation models. *arXiv preprint arXiv:2409.15370* (2024).

[148] Renxiao Wang, Xueliang Fang, Yipin Lu, Chao-Yie Yang, and Shaomeng Wang. 2005. The PDBbind database: methodologies and updates. *Journal of medicinal chemistry* 48, 12 (2005), 4111–4119.

[149] Weida Wang, Benteng Chen, Di Zhang, Wanhao Liu, Ben Gao, Shuchen Pu, Shuzhou Sun, Jin Zeng, Tianshu Yu, Wanli Ouyang, et al. 2026. Chem-r: Learning to reason as a chemist. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 12291–12302.

[150] Ziqing Wang and Kaize Ding. 2025. ReMol: LLM-guided Molecular Optimization with Reinforcement Learning. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (PhD Consortium). ACM*.

[151] David Weininger. 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences* 28, 1 (1988), 31–36.

[152] Mengsong Wu, YaFei Wang, Yidong Ming, Yuqi An, Yuwei Wan, Wenliang Chen, Binbin Lin, Yuqiang Li, Tong Xie, and Dongzhan Zhou. 2025. Chematagent: Enhancing llms for chemistry and materials science through tree-search based tool learning. *arXiv preprint arXiv:2506.07551* (2025).

[153] Yuyang Wu, Yue Huang, Shuaike Shen, Xujian Wang, Shuhao Zhang, Qiyao Xue, Weichen Liu, Runtian Gao, Jian Ma, Xiangliang Zhang, and Olexandr Isayev. 2026. Can Agents Price a Reaction? Evaluating LLMs on Chemical Cost Reasoning. *arXiv preprint arXiv:2605.07251* (2026). arXiv:2605.07251 https://arxiv.org/abs/2605.07251

[154] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. 2018. MoleculeNet: a benchmark for molecular machine learning. *Chemical Science* 9, 2 (2018), 513–530.

[155] Yutong Xie, Chence Shi, Hao Zhou, Yuwei Yang, Weinan Zhang, Yong Yu, and Lei Li. 2021. MARS: Markov molecular sampling for multi-objective drug discovery. In *International Conference on Learning Representations*.

[156] Qiang Xu, Shengyuan Bai, Leqing Chen, Zijing Liu, and Yu Li. 2025. ChemLabs on ChemO: A Multi-Agent System for Multimodal Reasoning on IChO 2025. *arXiv preprint arXiv:2511.16205* (2025).

[157] Zhiyuan Yan, Chen Liu, Boxuan Zhao, Kaiqing Lin, Jixiang Zhao, Yimi Wang, Liuzhenghao Lv, Hao Li, Shanzhuo Zhang, Li Yuan, et al. 2026. MoleCode unlocks structural intelligence in large language models. *arXiv preprint arXiv:2605.16480* (2026). https://arxiv.org/abs/2605.16480

[158] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, et al. 2019. Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling* 59, 8 (2019), 3370–3388.

[159] Yu Yang et al. 2024. Mol-AIR: Molecular reinforcement learning with adaptive intrinsic rewards. *Journal of Chemical Information and Modeling* (2024). doi:10.1021/acs.jcim.4c01669

[160] Zaifei Yang, Weiyu Chen, Yaqing Wang, and James Kwok. 2026. Probe Before You Edit: Probing-Guided Molecular Optimization for LLM Agents in Structure-Based Drug Design. *arXiv preprint arXiv:2606.00555* (2026).

[161] Zerui Yang, Yuwei Wan, Siyu Yan, Yudai Matsuda, Tong Xie, Bram Hoex, and Linqi Song. 2025. DrugMCTS: A drug repurposing framework combining multi-agent, RAG and Monte Carlo Tree Search. *arXiv preprint arXiv:2507.07426* (2025).

[162] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations*.

[163] Geyan Ye, Xibao Cai, Houtim Lai, Xing Wang, Junhong Huang, Longyue Wang, Wei Liu, and Xiangxiang Zeng. 2025. Drugassist: A large language model for molecule optimization. *Briefings in Bioinformatics* 26, 1 (2025), bbae693. doi:10.1093/bib/bbae693

[164] Xinwu Ye, Yicheng Mao, Jia Zhang, Yimeng Liu, Li Hao, Fang Wu, Zhiwei Li, Yuxuan Liao, Zehong Wang, Yingcheng Wu, et al. 2026. Latentchem: From textual cot to latent thinking in chemical reasoning. *arXiv preprint arXiv:2602.07075* (2026).

[165] Jiaxuan You, Bowen Liu, Rex Ying, Vijay Pande, and Jure Leskovec. 2018. Graph convolutional policy network for goal-directed molecular graph generation. *arXiv preprint arXiv:1806.02473* (2018).

[166] Botao Yu, Frazier N Baker, Ziru Chen, Garrett Herb, Boyu Gou, Daniel Adu-Ampratwum, Xia Ning, and Huan Sun. 2025. Tooling or not tooling? the impact of tools on language agents for chemistry problem solving. In *Findings of the Association for Computational Linguistics: NAACL 2025*. 7620–7640.

[167] Jiajun Yu, Yizhen Zheng, Huan Yee Koh, Shirui Pan, Tianyue Wang, and Haishuai Wang. 2025. Collaborative expert llms guided multi-objective molecular optimization. *arXiv preprint arXiv:2503.03503* (2025).

[168] Jie Yue, Bingxin Peng, Yu Chen, Jieyu Jin, Xinda Zhao, Chao Shen, Xiangyang Ji, Chang-Yu Hsieh, Jianfei Song, Tingjun Hou, et al. 2024. Unlocking comprehensive molecular design across all scenarios with large language model and unordered chemical language. *Chemical Science* 15, 34 (2024), 13727–13740.

[169] Atakan Yüksel, Erva Ulusoy, Atabey Ünlü, and Tunca Doğan. 2023. SELFormer: Molecular representation learning via SELFIES language models. *arXiv preprint arXiv:2304.04662* (2023).

[170] Barbara Zdrazil, Eloy Felix, Fiona Hunter, Emma J Manners, James Blackshaw, Sybilla Corbett, Marleen De Veij, Harris Ioannidis, David Mendez Lopez, Juan F Mosquera, et al. 2024. The ChEMBL Database in 2023: a drug discovery platform spanning multiple bioactivity data types and time periods. *Nucleic acids research* 52, D1 (2024), D1180–D1192.

[171] Huan Zhang, Yizhan Li, Wenhao Huang, Ziyu Hou, Yu Song, Xuye Liu, Farshid Effaty, Jinya Jiang, Sifan Wu, Qianggang Ding, et al. 2026. Towards Agentic Intelligence for Materials Science. *arXiv preprint arXiv:2602.00169* (2026).

[172] Jinwei Zhang, Xucheng Liang, Yu Zhang, Ruijie Yu, Xiaokang Yang, Yaohui Jin, and Yanyan Xu. 2026. ChemReason-Bench: Benchmarking Large Language Models for Procedural Reasoning in Experimental Chemistry. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 33211–33248. doi:10.18653/v1/2026.acl-long.1535

[173] Jia Zhang, Tengfei Ma, Tianle Li, Daojian Zeng, Xieping Gao, and Xiangxiang Zeng. 2026. Agents on a Tree: Pathwise Coordination for Multi-Objective Molecular Optimization. *arXiv preprint arXiv:2606.00008* (2026).

[174] Lisheng Zhang, Lilong Wang, Xiangyu Sun, Wei Tang, Haoyang Su, Yuehui Qian, Qikui Yang, Qingsong Li, Zhenyu Tang, Haoran Sun, Yingnan Han, Yankai Jiang, Wenjie Lou, Bowen Zhou, Xiaosong Wang, Lei Bai, and Zhengwei Xie. 2026. MolClaw: An Autonomous Agent with Hierarchical Skills for Drug Molecule Evaluation, Screening, and Optimization. *arXiv preprint arXiv:2604.21937* (2026). https://arxiv.org/abs/2604.21937

[175] Mingxu Zhang, Dazhong Shen, and Ying Sun. 2025. AtomDisc: An atom-level tokenizer that boosts molecular LLMs and reveals structure-property associations. *arXiv preprint arXiv:2512.03080* (2025).

[176] Wengyu Zhang, Xiao-Yong Wei, and Qing Li. 2026. Mol-Debate: multi-agent debate improves structural reasoning in molecular design. *arXiv preprint arXiv:2604.20254* (2026).

[177] Yu Zhang, Ruijie Yu, Jidong Tian, Feng Zhu, Jiapeng Liu, Xiaokang Yang, Yaohui Jin, and Yanyan Xu. 2025. ChemActor: Enhancing Automated Extraction of Chemical Synthesis Actions with LLM-Generated Data. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 24291–24314.

[178] Changmeng Zheng, Dayong Liang, Wengyu Zhang, Xiao-Yong Wei, Tat-Seng Chua, and Qing Li. 2024. A picture is worth a graph: A blueprint debate paradigm for multimodal reasoning. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 419–428.

[179] Andrew Y Zhou, Sharvaree Vadgama, Sumanth Varambally, Peter Eckmann, Michael K Gilson, and Rose Yu. 2026. ToolMol: Evolutionary Agentic Framework for Multi-objective Drug Discovery. *arXiv preprint arXiv:2605.12784* (2026).

[180] Zhenpeng Zhou et al. 2023. QADD: De novo drug design by iterative multi-objective deep reinforcement learning. *Bioinformatics* 39, 4 (2023), btad157. doi:10.1093/bioinformatics/btad157

[181] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N. Zare, and Patrick Riley. 2019. Optimization of molecules via deep reinforcement learning. *Scientific Reports* 9, 1 (2019), 10752. doi:10.1038/s41598-019-47148-x

[182] Yunheng Zou, Austin H Cheng, Abdulrahman Aldossary, Jiaru Bai, Shi Xuan Leong, Jorge Arturo Campos-Gonzalez-Angulo, Changhyeok Choi, Cher Tian Ser, Gary Tom, Andrew Wang, et al. 2025. El Agente: An autonomous agent for quantum chemistry. *Matter* 8, 7 (2025).