The Blind Spots of Agentic Drug Discovery

Edward Wijaya
StemRIM, Inc.
wijaya@stemrim.com

Abstract

Agentic systems have advanced drug discovery automation but reveal systematic blind spots beyond small-molecule workflows at well-resourced pharmaceutical companies. Drawing on experience with computational drug discovery projects at a small biotech specializing in therapeutic peptides, this perspective identifies five critical gaps: small molecule bias excluding protein language models and peptide-specific prediction; absent in vivo to in silico bridges for longitudinal, multi-modal animal data; LLM-centric orchestration that excludes ML training, reinforcement learning, and multi-paradigm coordination; resource assumptions mismatched to small biotech realities; and single-metric optimization ignoring multi-objective trade-offs in safety, efficacy, and stability. We propose design principles addressing each gap, including multi-paradigm orchestration, modality-aware architectures, in vivo integration, data-efficient learning, and Pareto-based optimization with uncertainty quantification. The goal is computational partners that augment practitioner judgment under realistic data, modality, and resource constraints.

1 Introduction: The Promise and the Blind Spots

1.1 The Current Narrative

Recent agentic AI systems have made tangible progress. Coscientist autonomously plans chemical syntheses [1], ChemCrow orchestrates 18 chemistry tools [2], and ChatInvent completed a 13-month deployment at AstraZeneca for literature synthesis [3]. PharmAgents integrates knowledge graphs for target identification [4], while MADD and DiscoVerse promise multi-agent collaboration. The dominant narrative positions agentic systems as the next frontier, moving beyond static models to systems that autonomously navigate literature, design experiments, and propose hypotheses [5, 6].

The architectural pattern is consistent: a large language model orchestrates tool calls, synthesizes results, and generates explanations. ChemCrow routes requests to RDKit, PubChem, and reaction prediction APIs. ChatInvent mines literature for research gaps. Coscientist interfaces with laboratory automation. This LLM-centric design works for text-based reasoning tasks: literature review, synthesis enumeration, protocol documentation, and safety analysis. The enthusiasm is warranted, but it is also narrowly scoped. Most systems are optimized for small-molecule workflows, high-throughput in vitro assays, and organizations with large datasets and extensive

compute. When those assumptions break, performance degrades in ways the demos do not reveal. This is the practical gap practitioners encounter in day-to-day work.

1.2 The Gap: What Happens Outside the Lab Automation Paradigm

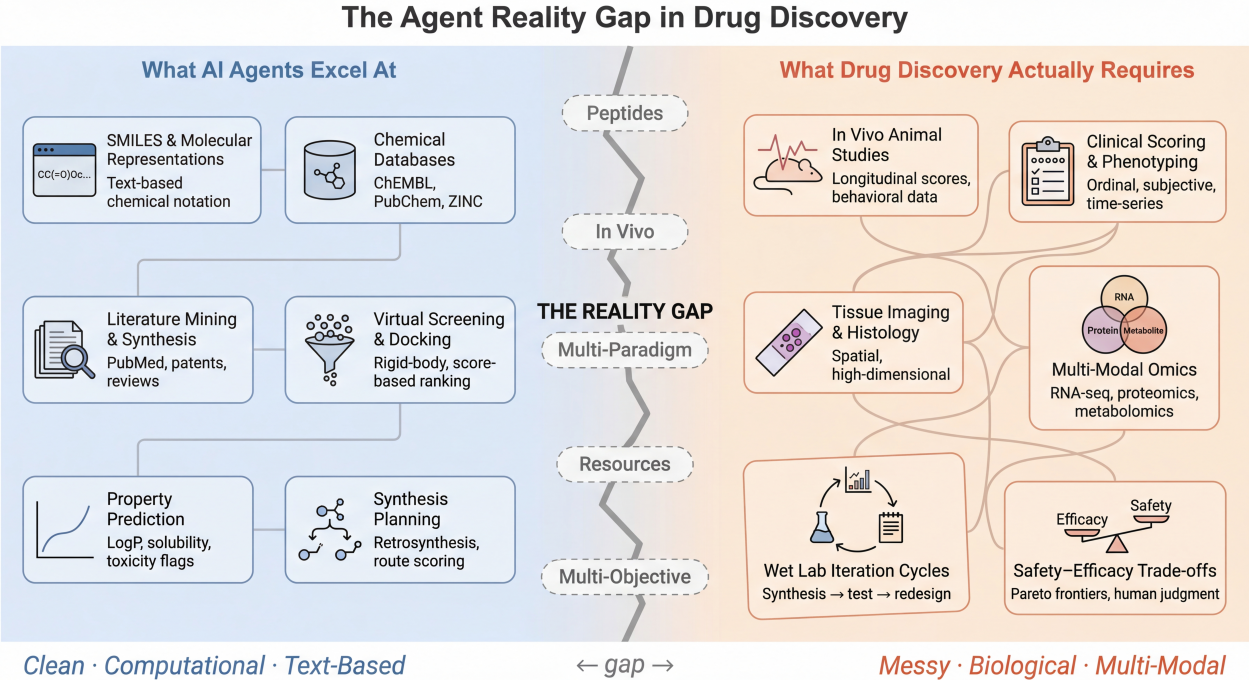

Figure 1: The Agent Reality Gap in Drug Discovery. Left panel shows computational workflows where current agents excel: small molecule representations (SMILES strings), databases, literature mining, and virtual screening. Right panel depicts the messy reality of drug discovery: multi-modal biological data from animal studies, wet lab iteration, and multi-objective trade-offs. The gap between these contexts represents the architectural limitations addressed in this paper.

However, these systems reveal systematic blind spots outside their design context: small-molecule discovery at well-resourced pharmaceutical companies. Peptide therapeutics require protein language models like ESM-2 [7] or ProtBERT [8], not molecular fingerprints. Peptides (5 to 50 amino acids) have complex conformational dynamics, aggregation propensities, and protease vulnerabilities absent in small molecules. No current agent supports protein language model fine-tuning, conformational sampling, or aggregation prediction.

In vivo efficacy studies generate longitudinal, multi-modal data: behavioral scores over weeks, tissue histology, RNA sequencing, and clinical notes. In traumatic brain injury models, efficacy manifests as motor coordination improvements at day 7, reduced neuroinflammation at day 14, and neurogenesis at day 28. No agent integrates these temporal data streams for outcome prediction. The result is a gap between in vitro promise and in vivo reality, where most development cost and risk actually sit.

Small biotechs face different constraints than AstraZeneca: 50 to 500 proprietary sequences

versus millions, single GPU versus clusters, one person handling design, modeling, and analysis. Transfer learning and few-shot adaptation are essential, not optional. Current agents assume abundant resources and long, interactive cycles that do not match small-team workflows.

Real drug discovery navigates multi-objective trade-offs under uncertainty. A peptide with tenfold higher bioactivity may have narrower safety margins or reduced stability. Current agents optimize single metrics or weighted sums, ignoring Pareto frontiers and uncertainty quantification. Practitioners end up doing this reasoning manually, which slows iteration and increases decision risk.

This paper draws on years of computational projects spanning peptide design, reinforcement learning optimization, in vivo efficacy modeling, behavioral phenotyping via computer vision, RNA-seq analysis, and multi-objective navigation. Each revealed architectural assumptions that do not generalize and where pragmatic workarounds were required to get results on real timelines. These experiences anchor the critique that follows. The following sections identify five critical gaps: small molecule bias (§2), absence of in vivo to in silico bridges (§3), LLM-centric orchestration limitations (§4), mismatch with small biotech realities (§5), and single-metric optimization (§6). We conclude with design principles for next-generation agents (§7).

2 Blind Spot 1: The Small Molecule Bias

Current agentic systems are architected around small molecules: SMILES strings, docking scores, synthetic accessibility metrics, and retrosynthesis planning. This works for medicinal chemistry but breaks down for peptide therapeutics requiring fundamentally different computational approaches.

2.1 The Peptide-Specific Challenge Space

Therapeutic peptides (2 to 50 amino acids) bridge small molecules and biologics. Unlike rigid small molecules, peptides are conformationally flexible, sampling diverse structural states. They achieve exquisite selectivity through induced-fit binding but face aggregation, protease degradation, and permeability barriers.

Peptide discovery diverges from small-molecule workflows. Structure-activity relationships do not transfer; conservative substitutions can abolish activity while drastic changes improve potency. Stability dominates. Developing peptides for traumatic brain injury, efficacy-stability trade-offs exceeded potency concerns. A bioactive peptide with minute-scale serum half-life has no therapeutic value. Protease resistance requires modeling interactions across dozens of enzyme families. Immunogenicity demands epitope scanning and MHC binding prediction. Aggregation depends on charge distribution and hydrophobic patterning.

Current agents provide no pathway for these requirements. ChemCrow includes RDKit, which does not handle peptide conformational sampling. PharmAgents assumes rigid-body docking, inappropriate for flexible peptides. ChatInvent mines small-molecule synthesis routes, irrelevant to peptide synthesis.

Practitioners encounter immediate friction. Peptides cannot encode as SMILES without losing stereochemistry. Molecular fingerprints (Morgan, MACCS) have no peptide analog. Rigid-molecule docking produces unreliable scores. Retrosynthesis metrics are meaningless. Even data storage becomes awkward: sequence variants, post-translational modifications, and assay metadata rarely fit small-molecule databases designed for single canonical structures.

2.2 Protein Language Models vs Molecular Fingerprints

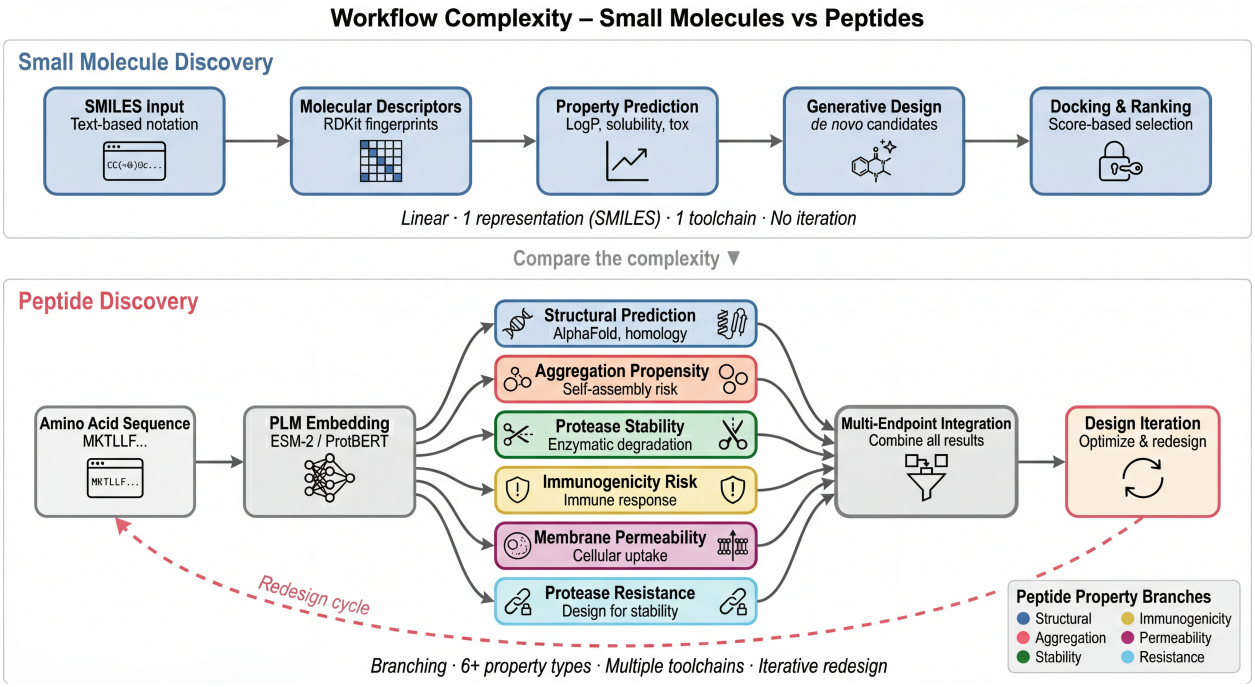

Figure 2: Workflow Complexity: Small Molecules vs Peptides. Top: Small molecule workflow follows a linear path from SMILES representation through RDKit property calculation to docking. Bottom: Peptide workflow branches into multiple parallel analysis streams including structural prediction, aggregation propensity, stability, immunogenicity, membrane permeability, and protease resistance, requiring integration of diverse computational tools and protein language models.

Protein language models define peptide design. ProtBERT [8], ESM-2 [7], and ProGen [9] encode evolutionary and structural priors from millions of protein sequences. ESM-2 embeddings predict receptor types from sequence alone. ProtBERT fine-tuning enables transfer learning with fewer than 100 examples.

Building peptide workflows requires capabilities current agents lack. Developing a receptor binding classifier involves curating training sets, extracting ESM-2 embeddings, training supervised classifiers, validating performance, and iterating hyperparameters. This is gradient-based ML, not API calls. The work also depends on small, noisy datasets where careful cross-validation and calibration matter more than single headline metrics. Agents must expose these uncertainties clearly so biologists can prioritize synthesis and testing. Without that, the workflow reverts to

manual triage and ad hoc heuristics. No agent executes this autonomously. LLM orchestrators assume models are black-box inference APIs. There is no fine-tuning support, dataset version control, or hyperparameter search. Agents retrieve embeddings but cannot adjust attention heads or train task-specific classifiers on proprietary data.

Generative modeling extends this gap. ProtGPT2 [10] fine-tunes on therapeutic sequences for de novo generation. Reinforcement learning optimizes multi-objective reward functions combining bioactivity and stability, requiring reward models, policy networks, gradients, and KL regularization. LLM agents cannot perform these ML workflows requiring end-to-end training control.

The gap reflects a missing paradigm, not a missing tool. Protein language models are the foundation of peptide discovery, demanding first-class support for training and fine-tuning. Without this, agents cannot support the core workflows practitioners use to move from sequence space exploration to experimentally validated leads.

2.3 What Peptide-Aware Agents Would Need

Peptide-aware architectures require protein language models as core components, not external APIs. First, fine-tuning pipelines: dataset curation, train-validation-test splits, learning rate scheduling, early stopping, checkpointing. Agents should fine-tune ProtBERT on 200 sequences and return calibrated classifiers with uncertainty estimates.

Second, structural biology integration. AlphaFold [11] structure prediction is central to peptide design. Flexible docking requires conformational sampling. Molecular dynamics provides stability and kinetics insights. These tools must integrate into multi-step workflows and feed back into sequence optimization, not sit as isolated analyses.

Third, multi-objective optimization. Peptide design balances bioactivity, stability, selectivity, and immunogenicity. We used curriculum learning for in vivo efficacy: initially rewarding bioactivity improvements, then progressively adding stability and toxicity constraints. This prevented local optima and maintained diversity. Current agents provide no framework for multi-stage optimization.

Fourth, diversity-aware generation. Generative models suffer mode collapse, producing reward-maximizing sequences with little variety. Practitioners use diversity penalties and max-min rewards, requiring state maintenance and dynamic sampling. LLM agents treat tool calls as stateless.

Finally, active learning loops. Limited budgets require careful peptide selection. Active learning maximizes information gain by prioritizing uncertain predictions. This requires uncertainty quantification, acquisition functions, and feedback loops updating models across multiple rounds.

Peptide discovery is a fundamentally different paradigm requiring ML training, structural biology, and multi-objective optimization as core capabilities. It also requires sequence-aware data management: tracking modifications, synthesis constraints, and assay provenance across iterative cycles. The small-molecule bias reflects architectural assumptions that must be revisited.

3 Blind Spot 2: The In Vivo to In Silico Bridge

The absence of in vivo modeling is a fundamental gap. Current agents excel at in vitro automation: Coscientist plans syntheses [1], ChemCrow screens compound libraries, ChatInvent mines literature. But critical validation happens in vivo, where candidates confront pharmacokinetics, biodistribution, metabolism, toxicology, and long-term efficacy unpredictable from binding affinity.

Animal studies generate fundamentally different data: longitudinal (days to months), multimodal (behavioral scores, imaging, molecular profiling), noisy (biological variability dwarfs plate assay precision), low-throughput (tens of compounds, not thousands), and expensive. These characteristics make in vivo the bottleneck, yet agents provide no pathway to incorporate this data.

3.1 The Lab Automation Ceiling

Lab automation reaches a hard ceiling at in vivo studies. Synthesis platforms and high-throughput screening test thousands of compounds daily. But animal experiments cannot be scaled or automated, requiring specialized facilities, personnel, ethical oversight, and weeks of time.

An agent might design a peptide and predict binding, but cannot integrate a 28-day traumatic brain injury study measuring behavioral recovery, histological regeneration, and transcriptomic neuroprotection. Data formats, temporal structure, and statistical requirements exceed current capabilities.

In vivo studies yield heterogeneous streams. Neurological injury evaluation includes behavioral assessments (motor coordination via beam walking, generating ordinal scores), tissue histology (cell proliferation requiring computer vision), RNA sequencing (high-dimensional gene expression needing differential expression and pathway enrichment), and clinical notes (semi-structured weight and adverse events). Current agents cannot integrate these modalities.

Developing composite efficacy metrics exposed this gap. A peptide increasing cell proliferation threefold in vitro shows temporal in vivo dynamics: inflammatory response (days 1-3), progenitor proliferation (days 7-10), functional recovery (day 28). Predicting sustained benefit requires temporal modeling, dataset curation, feature engineering, and validation outside LLM tool-calling scope.

3.2 Multi-Modal, Longitudinal Data Integration

The core challenge is that in vivo data does not fit the tidy CSV format that machine learning pipelines expect (Table 1). Behavioral scores are ordinal and subject to inter-rater variability.

Behavioral phenotyping via DeepLabCut [12] tracks animal poses in videos, generating time-series keypoint coordinates. Computing behavioral metrics (inter-animal distance, contact time, grooming) requires training pose estimation, validating tracking, computing features, and statistical testing. No agent executes this workflow spanning video processing, supervised learning, time-series engineering, and hypothesis testing.

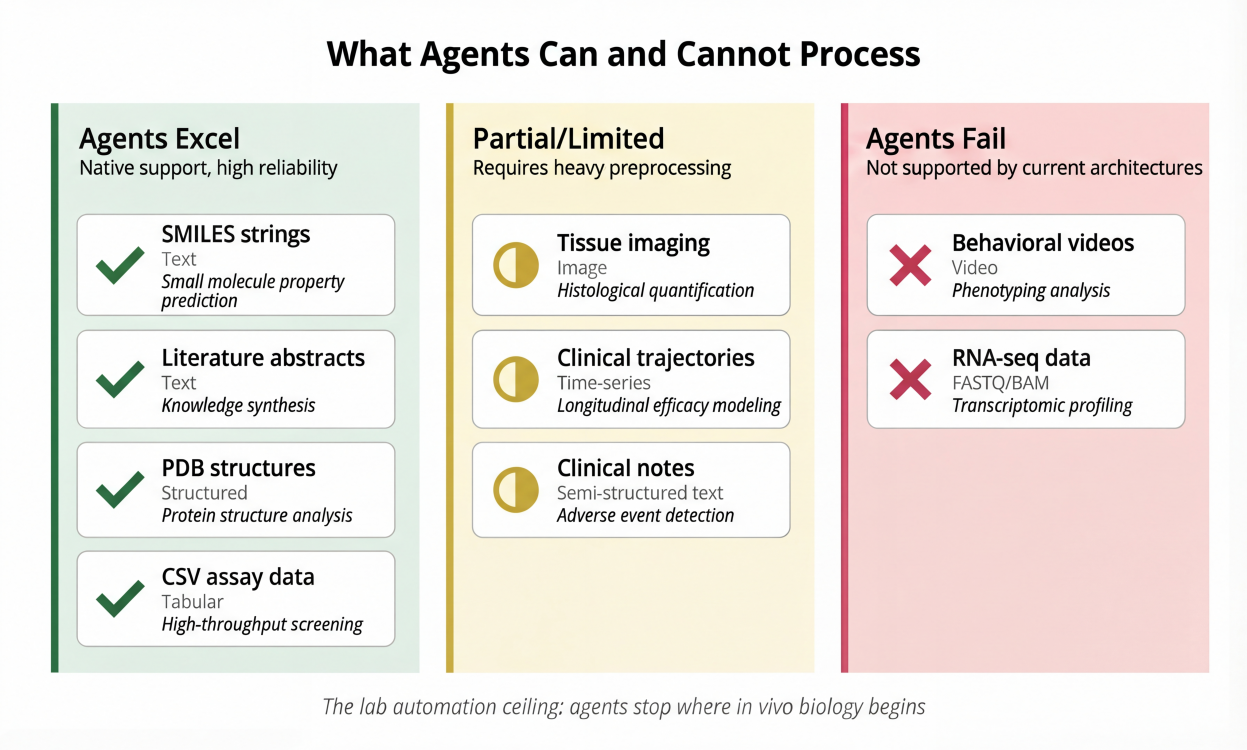

Figure 3: What Agents Can and Cannot Process. Matrix showing data types on the vertical axis and agent accessibility on the horizontal axis. Green checkmarks indicate data types current agents handle well (text, SMILES, PDB files, CSV data, literature). Red X marks denote data types agents struggle with (behavioral videos, clinical score trajectories, tissue imaging, multimodal transcriptomics, longitudinal measurements with dropout). This visualization reveals the systematic exclusion of in vivo data modalities from current agent architectures.

Table 1: Data Type Accessibility for Current Agent Systems

Data Type	**Format**	**Agent-Readable**	**Example Use Case**
SMILES strings	Text	Yes	Small molecule property prediction
Literature abstracts	Text	Yes	Knowledge synthesis
PDB structures	Structured	Yes	Protein structure analysis
CSV assay data	Tabular	Yes	High-throughput screening
Behavioral videos	Video	No	Phenotyping analysis
Clinical trajectories	Time-series	Partial	Longitudinal efficacy modeling
Tissue imaging	Image	Partial	Histological quantification
RNA-seq data	FASTQ/BAM	No	Transcriptomic profiling
Clinical notes	Semi-structured text	Partial	Adverse event detection

RNA-seq requires quality control, alignment, and quantification into expression matrices. Differential expression identifies treatment effects. Pathway enrichment maps genes to biological processes via KEGG [13] or Gene Ontology. Upstream regulator analysis infers transcription factors driving changes. The FASTQ-to-hypothesis pipeline needs bioinformatics tools (STAR, HISAT2, DESeq2, edgeR, GSEA [14]) that agents do not integrate.

Integrating heterogeneous sources for predictive biomarkers is valuable yet absent. Correlating in vitro bioactivity with in vivo efficacy required extracting features from multiple assays, normalizing across scales, aligning with temporal data (days 3, 7, 14, 28), and training regression models predicting long-term outcomes. This workflow involved feature engineering, imputation, stratified cross-validation, and model selection; these are ML workflows, not LLM reasoning.

3.3 Safety, Efficacy, and Translation

In vivo models surface safety-efficacy trade-offs that in silico screens miss. Tenfold bioactivity increases may trigger immune activation or hepatotoxicity. Stability modifications (D-amino acids, cyclization) may reduce affinity or increase aggregation. Optimal candidates balance objectives based on therapeutic context: high unmet need tolerates risk; chronic treatment demands safety.

Toxicology requires dose-response analysis via generalized linear mixed models accounting for repeated measures and time-dependent effects. Identifying therapeutic windows where efficacy plateaus but toxicity remains acceptable is essential. Agents cannot construct dose-response curves, compute LD50 confidence intervals, or visualize therapeutic indices.

Species translation compounds uncertainty. Mouse pharmacokinetics extrapolate to humans via allometric scaling. Peptide stability varies by species protease expression. Rodent-tolerated peptides may provoke primate antibody responses. Mechanistic modeling quantifying and propagating uncertainties into clinical predictions exceeds agent capabilities.

The most valuable support lies in uncertainty quantification, sensitivity analysis, and Pareto exploration, rather than autonomous decisions. Given candidates with efficacy and safety confidence intervals, which are Pareto-optimal? How sensitive are rankings to variability? Which experiments reduce uncertainty most? These require Bayesian optimization and value-of-information analysis beyond LLM tool-calling.

The in vivo to in silico bridge is a missing architectural layer for temporal modeling, multi-modal fusion, and mechanistic inference. Until agents ingest longitudinal scores, integrate transcriptomics, quantify dose-response uncertainty, and navigate multi-objective trade-offs under biological variability, utility remains confined to early hit identification. Most development cost and risk lies in translating in vitro activity to in vivo efficacy and safety.

4 Blind Spot 3: Multi-Paradigm, Not Multi-Agent

"Multi-agent" systems in drug discovery have multiple LLM-based agents collaborating [6]. PharmAgents deploys specialized agents for target identification and synthesis. MADD coordinates

molecular design and docking. The pattern: an orchestrator LLM decomposes queries, delegates to modules, and synthesizes outputs.

This distinction matters. "Multi-agent" is multiple LLM instances with different tools. Practitioners need multi-paradigm orchestration: coordinating fundamentally different computational approaches (supervised learning, generative modeling, RL, simulation, optimization) within workflows. Current architectures support the former, not the latter, which is why most real pipelines still require manual glue code.

4.1 The LLM-as-Orchestrator Assumption

LLM-centric design treats the language model as central coordinator, invoking external tools via APIs. ChemCrow calls RDKit and PubChem [2]. ChatInvent retrieves and synthesizes literature [3]. This works for text-based reasoning and stateless tools.

The paradigm breaks for tasks like training a multi-task neural network predicting peptide bioactivity across four endpoints using 300 sequences. The workflow: dataset preparation (stratified train-validation-test splits), feature extraction (ESM-2 embeddings), architecture selection, hyperparameter tuning, training with early stopping, validation with confidence intervals.

LLMs cannot orchestrate this via API calls. It requires gradient-based ML with end-to-end control over data loading, loss computation, parameter updates, and checkpointing. Agents do not support supervised learning as a first-class primitive they can configure, execute, monitor, and iterate.

The limitation extends beyond supervised learning. Generative modeling, reinforcement learning, Monte Carlo sampling, molecular dynamics, and Bayesian optimization all require iterative optimization with intermediate states, convergence monitoring, branching logic, resource management (GPU allocation, parallelization, checkpointing), and artifact versioning (models, hyperparameters, trajectories). None fit stateless API calls.

4.2 The Missing Paradigms

Absent paradigms define modern drug discovery: supervised learning (bioactivity prediction, toxicity modeling), unsupervised learning (chemical space clustering), generative modeling (de novo design), reinforcement learning (multi-objective optimization), simulation (binding kinetics), and optimization (experimental design).

Table 2: Agent Capability Matrix

Task Type	**Capability**	**Needs Human Review**	**Typical Runtime**
Literature review	✓	No	Minutes
SMILES generation	✓	Yes	Seconds
Docking (small molecules)	✓	Yes	Hours
Aggregation prediction	×	Yes	Days
In vivo analysis	×	Yes	Days to weeks
Multi-objective optimization	×	Yes	Hours to days

Across our projects, most computational work involved these paradigms, not LLM reasoning. Peptide-receptor classifiers required ESM-2 embeddings, logistic regression, gradient-boosted trees, cross-validation, and AUC-ROC comparison. RNA-seq needed alignment, normalization, differential expression, clustering, and pathway enrichment. Bone formation quantification trained semantic segmentation models. Peptide optimization via RL required reward models, proximal policy optimization, and diversity penalties.

Projects spanned paradigms in integrated pipelines: generative models produced sequences, supervised models predicted bioactivity, Bayesian optimization selected synthesis batches based on uncertainty, experimental results updated training sets, cycles repeated.

Agents cannot express these workflows. Orchestrators call models for inference but cannot train models, incorporate new data, retrain with hyperparameters, validate on test sets, or coordinate generative, predictive, and experimental design algorithms in closed loops. They assume pre-trained models and inference-only tasks, which is the opposite of how discovery actually proceeds.

4.3 What Multi-Paradigm Orchestration Looks Like

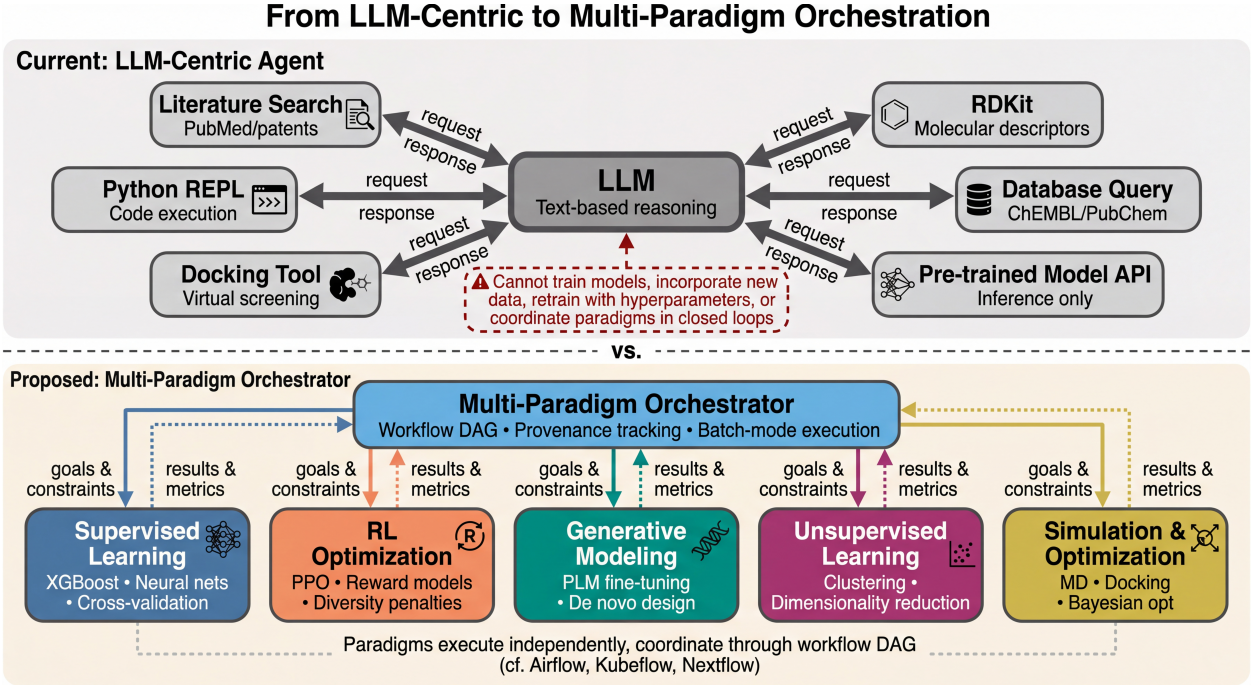

Figure 4: From LLM-Centric to Multi-Paradigm Orchestration. Top: Current LLM-centric architecture where a central language model orchestrates all tools through API calls. Bottom: Proposed multi-paradigm architecture where an orchestrator coordinates fundamentally different computational paradigms (ML training pipelines, RL optimization loops, PLM fine-tuning, CV analysis, physics simulations) that execute independently with results aggregated for decision-making.

Multi-paradigm architectures treat ML training, RL, simulation, and optimization as core primitives. Practitioners specify workflows declaratively: "Train ensemble bioactivity predictors (XG-

Boost, random forests, neural nets) with 5-fold cross-validation and hyperparameter tuning. Return Pareto frontier trading AUC-ROC versus calibration error." Agents translate specifications into executable workflow graphs, allocate resources, monitor convergence, and track provenance.

Workflow graphs have nodes (data loading, feature extraction, training, evaluation) and edges (data dependencies). They support parallelization (simultaneous hyperparameter configs), checkpointing (cached intermediate results), and branching (automatic hyperparameter search if validation fails), with clear provenance for every artifact. Workflow orchestration exists: Apache Airflow, Kubeflow, Nextflow provide task graphs, dependency resolution, resource allocation, and checkpointing. Missing is agent reasoning integration: inspecting results, diagnosing failures, proposing modifications, learning from executions (deprioritizing architectures that overfit).

Interaction should be batch-mode, not chat. Bottlenecks are orchestrating end-to-end analyses, not formulating queries. Small biotechs managing dozens of projects need automation: "Analyze eight RNA-seq samples, perform differential expression and pathway enrichment, generate report." Agents intervene only for human decisions (conflicting pathway interpretations).

Human-in-the-loop decision points are explicit and actionable. Agents present Pareto frontiers (accuracy-interpretability trade-offs), candidates with uncertainty estimates. Practitioners select based on context; agents structure decision spaces.

The gap reflects a mismatch with computational drug discovery practice. The field needs systems coordinating paradigms in integrated workflows, batch-mode execution, explicit decision points, and version control for reproducibility, not LLMs chatting.

5 Blind Spot 4: The Small Biotech Reality

ChatInvent's deployment at AstraZeneca [3] accessed institutional databases, HPC clusters, and proprietary libraries built over decades, with specialized teams (medicinal chemists, computational chemists, biologists, data scientists). This large pharma context defines current agent design assumptions but is not representative.

Small biotechs face different constraints: 50-100 employees, single wet labs, limited computational infrastructure, modest funding. Proprietary datasets have hundreds of compounds, not millions. One person designs experiments, analyzes results, and manages projects. Resource profiles are 10-100 times leaner, yet agents assume large pharma contexts.

5.1 Resource Constraints Current Agents Ignore

Data scarcity is the first constraint. Large pharma accumulates millions of tested molecules enabling high-capacity models. Small biotechs have 50-500 proprietary sequences. Every assay is precious.

Agents assume abundant data, recommending deep neural networks with millions of parameters, hundreds of hyperparameter configs, and dozens of ensemble models. On 100 sequences, deep networks overfit catastrophically. 5-fold cross-validation leaves only 20 examples for evaluation. Ensembles offer no benefit on small datasets.

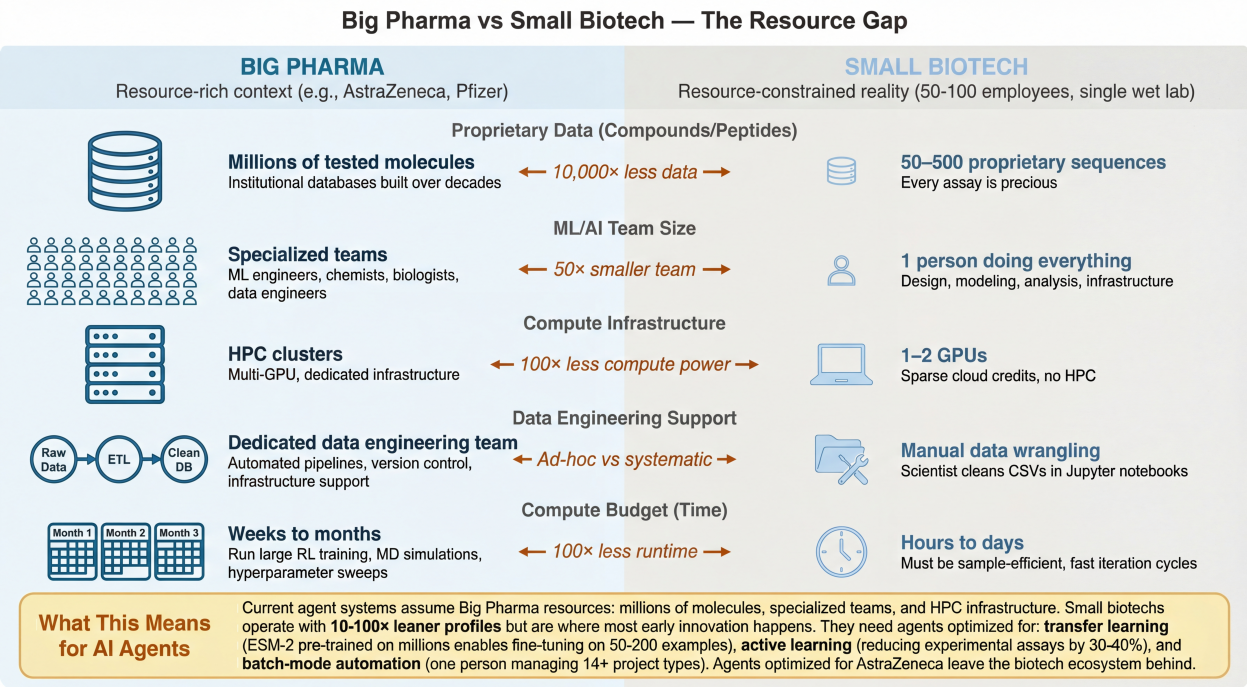

Figure 5: Big Pharma vs Small Biotech: The Resource Gap. Comparative visualization of computational resources, team size, and data availability. Left: Large pharmaceutical companies with 10,000+ compounds, 50-person ML teams, multi-GPU clusters, dedicated data engineers, and months of compute budget. Right: Small biotechnology companies with 50-200 compounds, 1-person computational teams, single GPU workstations, scientist-developers, and hours-to-days compute budgets. Current agent architectures assume the left context but significant innovation happens on the right.

Small biotechs need data efficiency: models generalizing from few examples via transfer learning, quantifying uncertainty for experimental design. ESM-2 pre-trained on millions of protein sequences enables fine-tuning lightweight classifiers on 50-200 examples. Active learning reduced assays 30-40

Computational constraints compound scarcity. Small biotechs have 1-2 GPUs, sparse cloud credits, no HPC. RL, molecular dynamics, and docking are expensive. Agents recommend intensive methods without considering infrastructure, ignoring efficiency optimizations (parallelization, caching, cheaper approximations). Resource-aware agents would propose: "Given one GPU and 24 hours: 100 high-precision docking runs or 1,000 fast approximations?"

Team structure is the third constraint. Large pharma has specialized roles (ML engineers, chemists, biologists, data engineers). Small biotechs have one person doing everything: peptide design, ML modeling, experimental analysis. No dedicated infrastructure support.

This demands tools for generalists handling infrastructure complexity (dependencies, environments, memory, debugging). Practitioners specify what, not how. Current agents assume infrastructure exists; generated code assumes installed libraries, formatted data, available resources. For small biotechs, these assumptions fail.

5.2 Transfer Learning and Few-Shot Adaptation

Transfer learning leverages public data for small proprietary tasks. Protein language models trained on UniProt's millions of sequences achieve performance with 50-200 examples that would require tens of thousands if trained from scratch.

Effective transfer learning requires selecting models (ESM-2, ProtBERT, ProGen), choosing fine-tuning layers (freezing early layers is data-efficient), setting learning rates (avoiding catastrophic forgetting or slow convergence), and implementing regularization. These are experimental decisions requiring domain knowledge.

Few-shot learning extends transfer to extreme scarcity. Prototypical networks achieved 60-70

Active learning selects which data to acquire next, prioritizing experiments reducing uncertainty. Acquisition functions (expected improvement, upper confidence bound) balance exploitation and exploration. Developing peptide bioactivity predictors, active learning reduced assays by one-third. Round one: 20 diverse peptides. Round two: 15 targeting high uncertainty. Round three: exploiting predicted best candidates. This iterative loop between models, acquisition functions, and feedback exceeds agent capabilities.

5.3 Batch-Mode Efficiency for Small Teams

Interactive chat assumes time for conversational interaction. This works for specific queries, not managing multiple projects simultaneously.

Small biotechs need batch-mode automation: "Analyze eight RNA-seq samples, identify differentially expressed genes, perform pathway enrichment, generate report." Agents execute autonomously overnight, intervening only for human decisions (which mechanism aligns with biological knowledge?).

Batch-mode requires robustness. If RNA-seq alignment fails (memory limits), agents should adjust parameters (reduce threads) or flag issues without losing progress. Checkpointing, fault tolerance, and version control enable reproducibility.

Parallelization is essential. Given 100 peptide docking jobs, agents should automatically parallelize across available resources (eight CPU cores, GPU acceleration), optimizing throughput without manual scheduling.

Current agents support minimal batch capabilities, designed for interactive queries not unsupervised analyses. They lack checkpointing, parallelization, and error handling, assuming interactive debugging impractical for overnight jobs.

Small biotech is not large pharma with fewer resources but a fundamentally different mode requiring data-efficient learning, transfer/few-shot adaptation, active learning, batch automation, and resource-aware optimization. Agents are optimized for AstraZeneca. The biotech sector where significant innovation happens is left behind.

6 Blind Spot 5: Multi-Objective Navigation

Drug discovery is multi-objective optimization: candidates must satisfy bioactivity, selectivity, safety, stability, manufacturability, and cost. Objectives conflict: potency improvements reduce selectivity, stability enhancements increase immunogenicity, high-purity synthesis is expensive. Navigating requires understanding Pareto frontiers (candidates where improving one objective degrades another) and decisions based on risk tolerance, development stage, and context.

Current agents optimize single objectives. ChemCrow optimizes binding affinity or synthetic accessibility [2]. Coscientist targets synthesis yield [1]. Multiple objectives collapse to weighted sums: "Maximize 0.6×bioactivity + 0.4×drug-likeness." This discards information: which candidates are Pareto-optimal, how sensitive are rankings to weights, what trade-offs exist. Agents present single "optimal" solutions, obscuring decision spaces.

6.1 The Single-Metric Trap

Single-metric optimization mirrors ML training objectives: minimize loss, maximize accuracy. This works for unidimensional goals, but drug discovery is multidimensional and context-dependent. Optimality depends on indication, development stage, competitive landscape, and risk tolerance. No scalar objective captures this.

Peptide stability illustrates the trap. A peptide that degrades rapidly in serum has no value. Stability modifications (D-amino acids, non-natural residues, cyclization) improve half-life but can reduce affinity, increase aggregation, or complicate synthesis. The balance depends on route of administration, therapeutic window, and development timeline.

In vivo peptide development, safety-efficacy trade-offs dominated selection. One peptide showed tenfold higher proliferation bioactivity but triggered hepatotoxicity at effective doses. Another had half the bioactivity but higher tolerated dosing, yielding comparable efficacy with better safety. A

third had intermediate bioactivity and safety but superior stability enabling less frequent dosing. "Optimal" depends on patient population, dosing, and regulatory risk tolerance.

Agents cannot represent these trade-offs. They predict A¿ B but cannot articulate: "A is twice as potent but has a threefold narrower safety margin; choose A if dosing can be tightly controlled, choose B for robustness." They do not visualize Pareto frontiers or sensitivity to weight changes.

6.2 Pareto Frontiers and Constraint Satisfaction

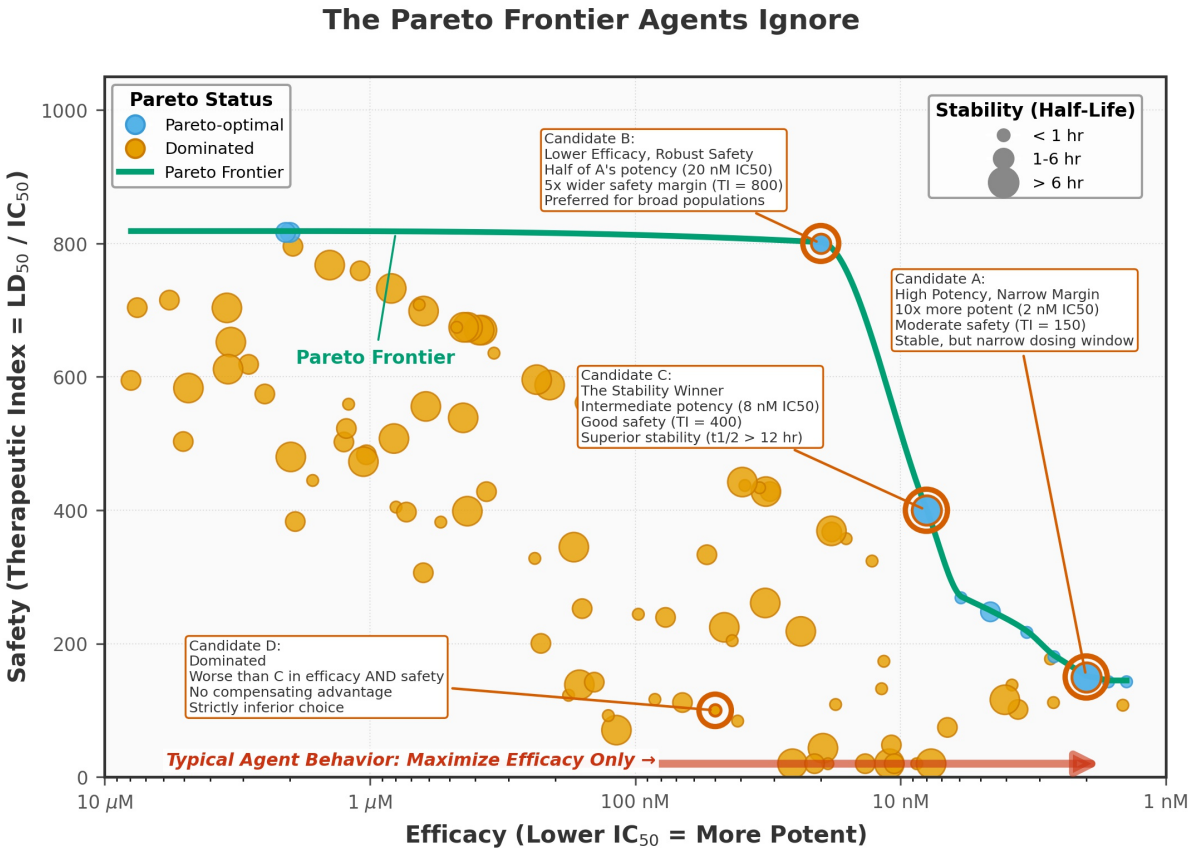

Figure 6: The Pareto Frontier Agents Ignore. Two-dimensional scatter plot of candidate compounds across efficacy (IC50) and safety (LD50 ratio), with stability (half-life) encoded by point size. The Pareto frontier curve identifies candidates where improving one objective requires degrading another. Annotations show real decision trade-offs: lower efficacy but much safer, highly effective but stability concerns. Current single-objective optimization (red arrow pointing to maximum efficacy) misses this complexity.

Pareto optimization is the appropriate framework. A candidate is Pareto-optimal if no other improves one objective without degrading another. The Pareto frontier is a curve (two objectives) or surface (three+). Practitioners navigate this frontier based on context.

Frontier visualization reveals trade-off structure. Steep regions require large sacrifices for modest gains; flat regions allow improvements with minimal cost. Clusters suggest distinct strategies (high-potency narrow-margin versus moderate-potency wide-margin). Gaps reveal unexplored regions.

Constructing the frontier requires multi-objective optimization. NSGA-II maintains candidate populations and selects non-dominated solutions. Multi-objective Bayesian optimization models objectives, selects candidates via acquisition functions balancing exploration and Pareto improvement, and updates with experimental results. These require tight integration between generative models, predictive models, and optimizers, which agents do not support.

Constraints add complexity: synthesizability, solubility, permeability, absence of toxicophores. Constrained optimization identifies the Pareto frontier within feasible regions, which may be disjoint or conflicting. In peptide design, synthesis feasibility is often binding. Sequences with non-natural residues may be predicted optimal but are unavailable or prohibitively expensive. Regioselective cyclization can be unfeasible. Practitioners must balance optimization with synthetic pragmatism.

6.3 Incorporating Uncertainty and Risk

Predictions include uncertainty. IC50 equals 10 nM might have a 95

Bayesian optimization handles uncertainty via Gaussian processes providing means and variances. Acquisition functions (expected improvement, upper confidence bound) balance exploitation and exploration. Over rounds, uncertainty shrinks in explored regions and stays high elsewhere. Agents should visualize uncertainty to guide exploration versus exploitation.

Risk profiles vary by stage. Early discovery tolerates high-risk, high-reward candidates. Late-stage demands well-characterized properties and high confidence. Agents should adapt recommendations accordingly.

In peptide pipelines, early rounds prioritized diversity, middle rounds balanced exploration and exploitation, final rounds focused on de-risking. This requires dynamic acquisition functions and explicit risk management absent from agents.

Sensitivity analysis is missing. How robust are rankings to model error? If bioactivity predictions have 20

Multi-objective navigation is the essence of decision-making. Agents must represent Pareto frontiers, quantify uncertainty, support risk-aware decisions, and structure decision spaces, not output single optima.

7 A Practitioner's Wishlist: What Would Actually Help

The gaps above reflect architectural assumptions that must be revisited. This section proposes five principles for next-generation agents grounded in peptide discovery, in vivo modeling, multi-paradigm workflows, resource constraints, and multi-objective decisions.

7.1 Design Principles for Next-Generation Agents

7.1.1 Principle 1: Multi-Paradigm Orchestration

Agents must support ML training, RL, simulation, and optimization as first-class primitives. Practitioners should specify workflows declaratively and receive an executable workflow graph that supports parallelization, checkpointing, and branching when validation fails. Human-in-the-loop decision points must be explicit when models trade off accuracy, calibration, or interpretability. Workflow orchestration systems (Airflow, Kubeflow, Nextflow) already provide these abstractions; what is missing is tight integration with agent reasoning and iterative improvement.

7.1.2 Principle 2: Modality-Aware Architectures

Agents must provide first-class support for peptides, proteins, and other modalities, not just small molecules. This requires PLM fine-tuning pipelines, structural biology integration (AlphaFold, flexible docking, molecular dynamics), and peptide-specific property prediction (aggregation, protease resistance, immunogenicity). Modality awareness must flow through data loaders, feature extractors, generative models, and evaluation metrics. Tool selection should be contextual: peptides default to ESM-2 embeddings, protein structures to structural alignment, not SMILES-based similarity.

7.1.3 Principle 3: In Vivo to In Silico Integration

Agents must support temporal modeling for longitudinal efficacy and safety data, multi-modal fusion (behavior, imaging, transcriptomics), and causal inference for mechanistic hypotheses. This requires bioinformatics pipelines (RNA-seq alignment, differential expression, pathway enrichment), computer vision, and statistical models for dose-response and mixed effects. Predictive efficacy modeling should align heterogeneous timepoints, handle missing data, and validate on temporally ordered splits. Mechanistic inference should integrate KEGG, GO, and Reactome to translate results into hypotheses and validation experiments.

7.1.4 Principle 4: Data Efficiency and Transfer Learning

Agents must optimize for few-shot adaptation, active learning, and knowledge transfer across related tasks. This requires meta-learning, Bayesian uncertainty quantification, and transfer pipelines combining public pre-training (UniProt, PDB, ChEMBL) with private fine-tuning. Agents should recommend strategies based on dataset size: deep nets for 500 examples, simpler models for 50, few-shot for 10. Active learning loops should select candidates via acquisition functions, update models with experimental feedback, and signal when marginal information gain is low.

7.1.5 Principle 5: Multi-Objective, Risk-Aware Optimization

Agents must support Pareto optimization with constraints, uncertainty quantification, sensitivity analysis, and interactive trade-off visualization. Results should be Pareto frontiers with confidence intervals, constraint status, and robustness scores. Practitioners should filter by feasibility, adjust objective weights, and highlight low-uncertainty candidates. Risk awareness should adapt recommendations to stage: early exploration versus late-stage de-risking.

7.2 Concrete Use Cases

These principles imply a shift from chat-first tooling to workflow-first systems. Agents should maintain state across iterations, log decisions, and make assumptions explicit so practitioners can audit results and reproduce outcomes. This is essential for regulated environments and for teams that revisit decisions months later, often under new personnel or budget constraints.

To make these principles concrete, we describe three representative use cases that current agents cannot handle but next-generation systems should support.

7.2.1 Use Case 1: Peptide Lead Optimization

Input: Fifty peptides with four assay endpoints, receptor structure, synthesis constraints.

Workflow: Fine-tune ESM-2, train multi-task regressor, use it as RL reward, filter by synthesis feasibility and stability, dock top candidates, cluster binding modes, present activity versus safety Pareto frontier with uncertainty.

Output: Ten synthesis candidates with rationales and confidence intervals.

Human decisions: Select among Pareto-optimal candidates based on strategic priorities and budget.

7.2.2 Use Case 2: In Vivo Efficacy Prediction

Input: In vitro assays and early in vivo markers for 20 peptides; predict day 28 outcomes.

Workflow: Normalize features, align temporal data with missing values, train regression models with stratified validation, identify early predictors and mechanistic drivers.

Output: Day 28 predictions with uncertainty and mechanistic hypotheses.

Human decisions: Choose peptides for full validation and whether to collect additional early markers.

7.2.3 Use Case 3: Multi-Endpoint Assay Analysis

Input: Four-endpoint data for 100 peptides.

Workflow: Normalize, cluster, validate stability, extract enriched sequence motifs, run pathway enrichment, visualize clusters.

Output: Distinct activity profile clusters with mechanistic hypotheses and selection recommendations.

Human decisions: Validate hypotheses and decide whether to focus on a single cluster or maintain diversity.

7.3 Infrastructure Needs

Table 3: Practitioner's Wishlist: Priority Matrix for Next-Generation Agent Features

Feature	Impact	Difficulty	Who Needs It	Priority
Multi-paradigm orchestration	High	High	All	Critical
PLM fine-tuning pipelines	High	Medium	Biologics	Critical
Active learning loops	High	Medium	Small biotech	Critical
Pareto frontier visualization	High	Low	All	High
Uncertainty quantification	High	Medium	All	High
In vivo data integration	Medium	High	All	High
Batch-mode workflows	Medium	Low	Small biotech	Medium
Transfer learning support	High	Medium	Small biotech	High
Constraint satisfaction	Medium	Medium	All	Medium
Workflow checkpointing	Low	Low	All	Medium

Realizing these capabilities requires infrastructure beyond current agents. API standards must cover PLMs (embedding extraction, fine-tuning, uncertainty), structural biology tools (AlphaFold, docking, molecular dynamics), and bioinformatics pipelines (alignment, differential expression, pathway enrichment). Version-controlled datasets, model registries, and provenance tracking are essential for reproducibility and auditability.

Organizational alignment matters. Systems must fit small biotech budgets with modest compute, minimal storage, and lightweight deployment. Documentation should target non-ML-expert biologists. Integration with existing tools (GraphPad, FlowJo, ImageJ, R/Bioconductor) avoids workflow disruption.

7.4 From Assistants to Partners

These systems should augment practitioner judgment, not replace it. Agents handle repetitive computation: preprocessing, training, tuning, orchestration, visualization. Practitioners focus on hypotheses, mechanistic interpretation, and strategic decisions under uncertainty.

Partnership requires bidirectional communication. Agents must explain reasoning, expose assumptions, and quantify uncertainty. Practitioners provide feedback and correct errors. Over time, agents learn which trade-offs matter and which models generalize.

Success is practitioner impact: faster lead identification, reduced experimental waste, better decisions under uncertainty. If agents embody these principles, they can transform workflows. If they remain LLM-centric tools designed for large pharma, they will not scale.

8 Conclusion: From Assistants to True Partners

Agentic drug discovery has reached a genuine inflection point. Systems like ChatInvent, Coscientist, and ChemCrow show that LLMs can orchestrate tools, navigate literature, and generate hypotheses. But current architectures are optimized for small-molecule discovery at well-resourced pharmaceutical companies, and blind spots appear outside that envelope.

This paper identified five gaps through practitioner experience across multiple projects: small-molecule bias limiting peptide and biologic work; absence of in vivo to in silico bridges for longitudinal, multi-modal data; LLM-centric orchestration excluding ML training, RL, simulation, and constrained optimization; large-pharma resource assumptions ignoring small biotech realities; and single-metric optimization missing multi-objective trade-offs under uncertainty. Addressing these gaps requires multi-paradigm orchestration, modality-aware architectures, in vivo integration, data-efficient learning, and Pareto-based optimization with uncertainty quantification.

These systems should not replace human expertise; drug discovery is too complex and context-dependent. The better framing is computational partners that handle preprocessing, model training, tuning, and visualization while practitioners focus on hypotheses, mechanistic interpretation, and strategic trade-offs. Partnership requires bidirectional learning: agents explain assumptions and uncertainty, practitioners provide feedback, and both improve over time.

Success is practitioner impact: faster lead identification, reduced experimental waste, and better decisions under uncertainty. If next-generation systems embody these principles, they can transform workflows. If they remain LLM-centric tools designed for large pharma, they will not scale. The difference is architectural, not cosmetic, and it will determine whether agents become core infrastructure or remain demonstrations. That choice will be visible in how teams allocate compute, structure experiments, and trust model outputs across discovery stages.

8.1 A Call to Action

For researchers: Move beyond tool-calling to multi-paradigm orchestration. Build data-efficient, modality-aware systems with uncertainty quantification and Pareto visualization. Prioritize batch-mode workflows over interactive chat for practitioners managing multiple projects.

For practitioners: Share anonymized problem formulations and dataset characteristics. Demand systems that respect resource constraints, quantify uncertainty, and support human-in-the-loop decision-making. Provide feedback on where agent recommendations succeed or fail.

For funders and organizations: Support open infrastructure for PLMs, structural biology tools, reproducible datasets, and workflow orchestration. Prioritize translational impact and cross-sector collaborations between large pharma and resource-constrained biotechs.

The past three years showed what is possible. The next three years will determine whether these systems generalize beyond demos. The gaps are engineering challenges. If the field meets them, agentic systems can become core infrastructure for drug discovery. If not, adoption will remain limited.

References

[1] Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624:570–578, 2023. doi: 10.1038/s41586-023-06792-0. Coscientist system for autonomous synthesis planning and execution.

[2] Andres M. Bran, Sam Cox, Andrew D. White, and Philippe Schwaller. ChemCrow: Augmenting large-language models with chemistry tools. *Nature Machine Intelligence*, 6:525–535, 2024. doi: 10.1038/s42256-024-00832-8. arXiv:2304.05376 (preprint April 2023). GPT-4 orchestrating 18 chemistry tools.

[3] Jiazhen He et al. Democratising real-world drug discovery through agentic AI. *Drug Discovery Today*, 31(2):104605, January 2026. doi: 10.1016/j.drudis.2026.104605. PMID: 41548711. ChatInvent system deployed at AstraZeneca.

[4] Bowen Gao, Yanwen Huang, Yiqiao Liu, Wenxuan Xie, Wei-Ying Ma, Ya-Qin Zhang, and Yanyan Lan. PharmAgents: Building a virtual pharma with large language model agents. *arXiv preprint arXiv:2503.22164*, March 2025. Multi-agent system for target identification and compound selection.

[5] Shaheen E. Lakhan. The agentic era: Why biopharma must embrace AI that acts. *Cureus*, May 2025. doi: 10.7759/cureus.83390. PMID: 40322603. Editorial on agentic AI in pharmaceutical industry.

[6] Srijit Seal et al. AI agents in drug discovery. *arXiv preprint arXiv:2510.27130*, October 2025. Comprehensive 45-page survey of agentic AI systems.

[7] Zeming Lin, Halil Akin, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574. PMID: 36927031.

[8] Ahmed Elnaggar et al. ProtTrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10): 7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381. Online July 2021, print October 2022.

[9] Ali Madani et al. Large language models generate functional protein sequences across diverse families. *Nature Biotechnology*, 41:1099–1106, 2023. doi: 10.1038/s41587-022-01618-2.

[10] Noelia Ferruz, Steffen Schmidt, and Birte Höcker. ProtGPT2 is a deep unsupervised language model for protein design. *Nature Communications*, 13:4348, 2022. doi: 10.1038/s41467-022-32007-7.

[11] John Jumper et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596: 583–589, 2021. doi: 10.1038/s41586-021-03819-2. PMID: 34265844.

[12] Alexander Mathis et al. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21:1281–1289, 2018. doi: 10.1038/s41593-018-0209-y. PMID: 30127430.

[13] Minoru Kanehisa, Miho Furumichi, Yoko Sato, Masayuki Kawashima, and Mari Ishiguro-Watanabe. KEGG for taxonomy-based analysis of pathways and genomes. *Nucleic Acids Research*, 51(D1):D587–D592, 2023. doi: 10.1093/nar/gkac963. PMID: 36300620.

[14] Aravind Subramanian et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005. doi: 10.1073/pnas.0506580102. PMID: 16199517.