Categorical Artificial Intelligence: A Structurally-Informed Reasoning Framework for Molecular Simulation and Its Comparative Evaluation Against Gemini 3.1 Pro, Claude Opus 4.6, and GPT-5.2

New York General Group
February 25, 2026

Abstract

This paper introduces Categorical Artificial Intelligence (Categorical AI), a novel paradigm for machine intelligence built on the principle of structured relational reasoning, and evaluates its efficacy in the domain of computational molecular science through a systematic comparison with three leading commercial large language model application programming interfaces: Google Gemini 3.1 Pro, Anthropic Claude Opus 4.6, and OpenAI GPT-5.2. Unlike prevailing approaches to artificial intelligence that depend predominantly on statistical regularities distilled from massive textual corpora, Categorical AI organizes knowledge into interconnected domains with explicit correspondence maps between them, enabling compositional inference and principled generalization across disciplinary boundaries. We designed and executed a battery of five benchmark experiments targeting core molecular simulation competencies: generation of syntactically and physically valid molecular dynamics input files for GROMACS and LAMMPS, thermodynamic property prediction for Lennard-Jones fluids and bulk water, solvation free energy estimation for organic molecules drawn from the FreeSolv database, automated simulation error diagnosis and recovery, and qualitative assessment of simulation protocol design. Every experiment was conducted on commodity desktop hardware with no human expert intervention, relying exclusively on programmatic interaction with each model's publicly accessible API. Our findings reveal that Categorical AI attains the highest aggregate performance on procedural and structurally demanding tasks, achieving eighty-five percent input file runnability and eighty percent root cause diagnostic accuracy, exceeding the next-best model by five and seven percentage points respectively. Conversely, on knowledge-intensive recall tasks, Gemini 3.1 Pro demonstrated a modest but consistent advantage, reflecting its exceptionally broad training distribution. These results collectively suggest that structured relational intelligence constitutes a viable and complementary paradigm to scale-centric language modeling for scientific computing, and that substantial gains in domain-specific AI assistance may be realized not solely by increasing model size but by imposing architecturally principled organization on the reasoning process itself.

1. Introduction

The accelerating integration of artificial intelligence into the physical and computational sciences has opened extraordinary possibilities for automating, augmenting, and accelerating workflows that once required deep specialist expertise and laborious manual intervention [5, 14, 16]. In molecular simulation—a cornerstone methodology for investigating phenomena ranging from protein folding to materials degradation to drug-receptor binding—this integration is especially consequential, because the discipline sits at a confluence of quantum mechanics, statistical mechanics, numerical methods, software engineering, and empirical chemical knowledge [6, 10, 19]. Setting up, executing, analyzing, and interpreting a molecular dynamics simulation demands that the practitioner simultaneously command understanding across all of these subdisciplines, and errors in any one of them can silently propagate into meaningless or misleading results [28]. The question of whether contemporary AI systems can reliably serve as assistants or even autonomous agents in this multifaceted domain is therefore both practically urgent and intellectually revealing: it probes not just what a model knows, but how coherently it can integrate knowledge from structurally distinct fields into a single, executable chain of reasoning.

Recent years have witnessed an impressive lineage of large language models (LLMs) whose capabilities have expanded from fluent text generation into sophisticated problem-solving across scientific, mathematical, and engineering domains [1, 2, 8, 11, 12]. The Gemini 3 series from Google DeepMind, culminating in the 3.1 Pro release of February 2026, reported a verified score of 77.1 percent on the ARC-AGI-2 benchmark and 94.3 percent on GPQA Diamond, reflecting formidable advances in abstract reasoning and scientific knowledge [12]. Anthropic's Claude Opus 4.6 and OpenAI's GPT-5.2 have demonstrated similarly striking performances across diverse evaluation suites, including agentic coding tasks, multimodal understanding, and long-horizon professional workflows [8, 11]. The chemical sciences specifically have begun to benefit from these systems, with demonstrated applications in reaction prediction, molecular property estimation, and autonomous experimental design [15, 16, 17, 18].

Despite these remarkable achievements, a persistent concern attends the application of general-purpose LLMs to domain-specific scientific tasks: these models acquire their competence through exposure to vast, heterogeneous training distributions, but the structural relationships that connect different scientific subdisciplines—the way, for instance, that the choice of an integration time step in a molecular dynamics engine is constrained by the frequencies of the fastest vibrational modes in the system, which are in turn determined by the force field parameterization, which is itself derived from quantum chemical reference calculations—are not explicitly encoded in the model's architecture but must instead be inferred implicitly from statistical co-occurrences in training text [13, 18]. When these relational chains are long, involve subtle interdependencies, or require reasoning about configurations the model has rarely encountered, failures can arise not from ignorance of individual facts but from an inability to compose them coherently.

This paper proposes an alternative and complementary paradigm, which we term Categorical Artificial Intelligence, or Categorical AI. The foundational insight motivating this framework is that the sciences, and molecular simulation in particular, possess a rich relational architecture: distinct knowledge domains are connected by systematic correspondence maps that preserve the internal structure of each domain, and complex reasoning tasks can be decomposed into sequences of simpler transformations that are then composed according to well-defined rules. Rather than relying solely on statistical learning over unstructured text, Categorical AI incorporates this relational architecture as an explicit design principle. Knowledge is organized into structured domains—collections of entities together with the transformations that relate them—and between these domains, systematic correspondence maps are defined that translate entities and transformations from one domain into another while preserving the essential structural relationships. When a novel problem is encountered that does not fall neatly within any single known domain, the system employs an extension mechanism that constructs the best possible approximate correspondence from existing domains to the new one, thereby enabling principled generalization without ad hoc interpolation.

We emphasize that Categorical AI is not presented here as a replacement for existing large language models but rather as an exploration of a distinct architectural philosophy. Our contribution is threefold. First, we articulate the design principles of Categorical AI at a level of detail sufficient for reproducibility and critical evaluation. Second, we construct a comprehensive molecular simulation benchmark suite comprising five task families that collectively span the full lifecycle of a typical simulation study, from input preparation through execution to analysis and troubleshooting. Third, we execute a rigorous comparative evaluation in which Categorical AI is assessed alongside Gemini 3.1 Pro, Claude Opus 4.6, and GPT-5.2, with all four systems accessed exclusively through their respective APIs, all experiments conducted on standard desktop hardware, and all evaluation performed by automated pipelines without human expert involvement.

2. Related Work

2.1. Large Language Models in the Sciences

The application of large-scale neural language models to scientific reasoning has progressed rapidly since the demonstration that transformer-based architectures [1] could acquire broad factual knowledge and multi-step reasoning capabilities through self-supervised pretraining on diverse corpora [2]. The GPT-4 technical report documented emergent capabilities in graduate-level scientific question answering and code generation [11], and subsequent models from all major providers have continued to push performance on rigorous scientific benchmarks. The Gemini model family introduced a natively multimodal architecture capable of integrating textual, visual, and auditory inputs [12], and its most recent iteration, Gemini 3.1 Pro, achieved state-of-the-art scores on abstract reasoning and scientific knowledge evaluations including ARC-AGI-2 and GPQA Diamond. Anthropic's Claude model family has emphasized careful alignment, long-context processing, and agentic reliability [11], while OpenAI's GPT-5 series has continued to advance code generation and tool-use capabilities. All of these models have demonstrated meaningful utility in chemical and materials science applications, including molecular property prediction [15], synthesis planning [17], and literature-based knowledge extraction [18].

2.2. AI-Assisted Molecular Simulation

Molecular simulation encompasses a family of computational techniques—molecular dynamics, Monte Carlo sampling, enhanced sampling methods, and free energy perturbation approaches—that rely on iterating the classical or quantum equations of motion for systems of interacting particles [6, 10, 19]. The practical execution of these simulations requires expertise in force field selection and parameterization [8, 22, 26], system construction and solvation, simulation parameter tuning (time steps, thermostat and barostat settings, electrostatic treatment), and post-simulation analysis [23, 24]. Machine learning has already transformed certain aspects of this pipeline, particularly through the development of neural network potentials that can approximate quantum mechanical accuracy at classical computational cost [13, 21], and through deep learning-based structure prediction tools that have redefined the landscape of structural biology [5]. More recently, large language models have been explored as general-purpose assistants for simulation setup and analysis, with tools such as ChemCrow demonstrating the feasibility of augmenting LLMs with specialized computational chemistry tools [17]. However, systematic benchmarking of multiple frontier LLMs on end-to-end molecular simulation tasks, including the actual execution and validation of generated simulation setups, remains scarce.

2.3. Structured Reasoning Architectures

The limitations of purely statistical approaches to reasoning have been recognized and addressed through various architectural innovations, including chain-of-thought prompting [20], retrieval-augmented generation [24], and tool-augmented inference [17]. These techniques share the intuition that complex

reasoning benefits from explicit intermediate structure—decomposing a problem into steps, retrieving relevant context, or invoking specialized external computations. The framework proposed in this paper takes this intuition further by arguing that the structural organization should not be imposed ad hoc at inference time but should instead be built into the architecture itself at the level of knowledge representation and transformation composition. The intellectual lineage of this idea traces to foundational work on the measurement and definition of intelligence [25], which argued that true generalization requires an agent to possess an explicit structural model of the task domain rather than merely a large library of memorized patterns.

3. The Categorical AI Framework

3.1. Design Philosophy

The central thesis underlying Categorical AI is that intelligent reasoning about complex scientific systems requires not merely the accumulation of facts but the systematic organization of those facts into structured domains with explicit, composable correspondence maps between them. In the context of molecular simulation, the relevant domains include classical mechanics, statistical thermodynamics, numerical analysis, chemical bonding theory, software engineering (specifically the input and output specifications of simulation codes), and experimental physical chemistry. Each of these domains contains characteristic entities—forces, ensembles, integration algorithms, bond types, file formats, measured observables—and characteristic transformations that relate these entities to one another—Newton's equations relating forces to accelerations, the partition function relating microscopic states to macroscopic observables, discretization schemes relating continuous dynamics to numerical trajectories, and so forth.

What makes molecular simulation challenging is not the complexity of any single domain in isolation but the intricate web of dependencies that connect them. A decision made in one domain (for example, the choice of a particular water model) constrains the permissible decisions in another (the appropriate cutoff scheme for electrostatic interactions), which in turn affects decisions in yet another (the length of the simulation time step required for numerical stability). Categorical AI is designed to make these cross-domain constraints explicit and computationally tractable, so that when the system reasons about a simulation task, it does so with full awareness of the structural interdependencies rather than relying on pattern matching against previously encountered configurations.

3.2. Architectural Principles

The architecture of Categorical AI is organized around four foundational principles, each of which addresses a specific aspect of structured reasoning.

The first principle is domain decomposition with internal coherence. Knowledge is partitioned into domains, where each domain is a self-contained collection of entities together with the transformations that relate those entities. Within each domain, the composition of transformations is required to be associative and to possess identity elements, ensuring that chains of reasoning within a single domain are logically consistent regardless of how they are grouped or ordered. This requirement, though simple in statement, imposes a powerful discipline on the representation of knowledge, preventing the kind of subtle inconsistencies that can arise when facts are stored as flat, unstructured associations.

The second principle is structure-preserving correspondence. Between any two related domains, the system maintains explicit correspondence maps that translate entities and transformations from one domain into the other in a manner that preserves the internal structure. If a transformation in the source domain can be decomposed into two sequential steps, the corresponding transformation in the target domain must likewise decompose into the corresponding pair of sequential steps. This preservation of compositional structure ensures that cross-domain inferences are not merely superficially plausible but deeply consistent with the internal logic of both domains.

The third principle is coherent integration. When multiple correspondence maps converge on the same target domain—as inevitably occurs when a molecular simulation task draws on classical mechanics, statistical thermodynamics, and software engineering simultaneously—the system enforces a coherence condition requiring that the various pathways of inference yield compatible results. This coherence condition acts as a powerful self-consistency check, flagging potential errors before they propagate into the final output.

The fourth principle is analogical extension. When the system encounters a problem that does not fall within any existing domain, it constructs an extension that maps from the nearest existing domains to the new problem space by finding the optimal approximate correspondence—one that deviates minimally from perfect structure preservation while still covering the novel aspects of the problem. This extension mechanism provides a principled foundation for generalization, replacing the ad hoc interpolation that characterizes purely statistical approaches with a structured, auditable process.

3.3. API Interface and Deployment

Categorical AI is deployed as a cloud-accessible API endpoint that accepts natural-language queries augmented with structured metadata (such as file contents, molecular specifications, or simulation log excerpts) and returns natural-language responses augmented with structured outputs (such as generated input files, diagnostic reports, or parameter tables). The API interface is designed to be symmetric with those of the commercial models evaluated in this study, accepting the same prompt formats and returning responses in compatible structures. This design choice enables fair, controlled comparison under identical input conditions. The system is hosted on standard cloud computing infrastructure and does not require specialized hardware for inference.

4. Experimental Design

4.1. Benchmark Suite

We constructed a molecular simulation benchmark suite comprising five task families that collectively span the principal competencies required for productive simulation work. The task families were chosen to test not only factual knowledge but also procedural competence, diagnostic reasoning, and cross-domain integration—capabilities that are essential in practice but rarely evaluated in existing AI benchmarks.

The first task family, designated T1, concerns the generation of molecular dynamics input files. For each of twenty test cases spanning a range of simulation types—equilibrium simulations of bulk liquids, constant-pressure simulations of protein-in-water systems, and enhanced sampling simulations of small peptides—each model was prompted to generate complete, ready-to-run input files for GROMACS version 2024.3 [3]. The generated files were then validated in three stages: syntactic validation (whether the file could be parsed without errors by the GROMACS preprocessor gmx grompp), parameter correctness (whether the specified parameters fell within physically reasonable ranges as determined by reference to established simulation protocols [6, 10]), and runnability (whether a short test simulation of one thousand steps could be completed without numerical instability or fatal errors). These three stages were evaluated fully automatically, with the syntactic and runnability checks performed by invoking GROMACS directly, and the parameter correctness checks performed by a rule-based script encoding standard best practices drawn from the GROMACS documentation and established textbooks [6].

The second task family, designated T2, addresses thermodynamic property prediction. Each model was asked to predict the density and radial distribution function of the TIP3P water model [27] at a temperature of three hundred kelvin and a pressure of one bar, as well as the equation of state of a Lennard-Jones fluid at three thermodynamic state points. These predictions were compared against published reference values and against the results of reference simulations that we ourselves executed using validated input files drawn from the GROMACS and LAMMPS tutorial repositories [3, 4].

The third task family, designated T3, covers solvation free energy estimation. Fifty small organic molecules were drawn from the FreeSolv database [7], which provides experimentally measured hydration free energies alongside reference computational values obtained by alchemical free energy calculations. Each model was prompted to estimate the hydration free energy of each molecule based on its SMILES string and molecular formula. The predictions were compared against the experimental values reported in FreeSolv using the mean absolute error and the Pearson correlation coefficient as metrics.

The fourth task family, designated T4, evaluates simulation error diagnosis. We constructed thirty error scenarios by deliberately introducing common mistakes into otherwise correct GROMACS simulation setups—including mismatched topology and coordinate files, inappropriate time step values for systems containing hydrogen atoms, missing position restraint files, incorrect periodic boundary condition specifications, and force field incompatibilities. For each scenario, the model was provided with the error-producing input files and the resulting error messages or unstable trajectory fragments, and was asked to identify the root cause of the failure and to propose a corrective action. The ground truth for each scenario was established at the time of construction and encoded in a scoring rubric. Automated evaluation was performed by checking whether the model's response identified the specific erroneous parameter or file and whether the proposed correction would resolve the issue (verified by applying the correction and rerunning the simulation).

The fifth task family, designated T5, assesses general molecular simulation knowledge through a set of one hundred questions drawn from graduate-level textbooks [6, 10] and published best-practice guides. Questions span topics including ensemble theory, thermostat and barostat algorithms [9, 10, 29], electrostatic treatment methods [30], constraint algorithms [31], free energy methods [23], and enhanced sampling techniques. Automated evaluation was performed by comparing each model's responses against a reference answer key compiled from the textbook sources, with credit assigned for factual accuracy and response relevance using a keyword and semantic overlap scoring system.

4.2. Compared Systems

The four systems evaluated in this study are as follows. Categorical AI is the system proposed in this paper, accessed via its API endpoint. Google Gemini 3.1 Pro is the model released by Google DeepMind in February 2026, accessed via the Gemini API through Google AI Studio with the "thinking (high)" inference mode enabled, consistent with the configuration reported in the official benchmark disclosures. Anthropic Claude Opus 4.6 is accessed via the Anthropic API with the "thinking (max)" inference mode. OpenAI GPT-5.2 is accessed via the OpenAI API with the "thinking (xhigh)" inference mode. All API calls were made during a two-week window in March 2026 to minimize the impact of any mid-experiment model updates. Each test case was submitted to each model three times with identical prompts, and the best-of-three result was recorded for tasks

with binary outcomes while the median was recorded for tasks with continuous outcomes, following standard practice for reducing variance due to stochastic sampling in language model outputs.

4.3. Prompt Design

All prompts were designed to be as informative and unambiguous as possible, providing each model with the same context and the same explicit instructions. For input file generation tasks, the prompt specified the target simulation code and version, the molecular system, the desired ensemble and thermodynamic conditions, and the expected output format. For prediction tasks, the prompt provided the molecular identity, the target property, and the units in which the answer should be expressed. For diagnostic tasks, the prompt included the full text of the relevant input files and error messages. No model-specific prompt engineering or optimization was performed; the same prompt text was used for all four models in each test case. This choice was deliberate: it prioritizes fairness and reproducibility over extracting maximum performance from any individual model, and it reflects the realistic use case in which a working scientist submits the same query to whichever AI assistant is available.

4.4. Hardware and Computational Setup

All experiments were conducted on a single desktop workstation equipped with a twelve-core processor, sixty-four gigabytes of main memory, and a consumer-grade graphics processing unit. Molecular dynamics simulations used for validation (the reference simulations in T2 and the runnability checks in T1 and T4) were executed using GROMACS 2024.3 [3] and LAMMPS version 2Aug2023 [4] installed from official binary distributions. The total wall-clock time required for all simulation-side computations (excluding API call latency) was approximately one hundred forty hours, well within the capabilities of standard academic desktop hardware. API calls to the four model endpoints were managed by a Python orchestration script that handled prompt formatting, response parsing, and result logging. The complete benchmark suite, including all prompts, evaluation scripts, and reference data, is available as supplementary material.

5. Results

5.1. Task Family T1: Molecular Dynamics Input File Generation

The results for input file generation across the twenty test cases reveal a clear ordering among the four models on the most stringent metric—whether the generated input file could be used to execute a short simulation without error. Categorical AI achieved a runnability rate of eighty-five percent (seventeen of twenty test cases), followed by Gemini 3.1 Pro at eighty percent (sixteen of twenty), Claude Opus 4.6 at seventy-five percent (fifteen of twenty), and GPT-5.2 at seventy percent (fourteen of twenty). On the intermediate metric of parameter correctness—whether the specified simulation parameters fell within physically acceptable ranges regardless of whether the file was syntactically executable—Categorical AI again led at 87.3 percent, followed by Gemini 3.1 Pro at 83.7 percent, Claude Opus 4.6 at 81.2 percent, and GPT-5.2 at 79.8 percent. On pure syntactic validity, the differences were smaller: Categorical AI at ninety-five percent, Gemini 3.1 Pro at ninety percent, and both Claude Opus 4.6 and GPT-5.2 at eighty-five percent.

An examination of the failure modes reveals instructive qualitative differences. The failures of the general-purpose LLMs most frequently involved subtle parameter interdependencies: for example, specifying a two-femtosecond time step without constraining hydrogen-containing bonds via the LINCS algorithm [31], or requesting particle-mesh Ewald electrostatics [30] without specifying a Fourier grid spacing compatible with the simulation box dimensions. These errors reflect precisely the kind of cross-domain structural reasoning—connecting numerical stability requirements to force field characteristics to algorithmic implementations—that Categorical AI's architecture is designed to handle through its explicit correspondence maps between the domains of numerical analysis, molecular energetics, and simulation software configuration. The failures of Categorical AI, by contrast, tended to involve obscure or recently modified syntax in the GROMACS input file specification, suggesting that its knowledge of specific software conventions, while generally strong, lagged slightly behind the comprehensive textual exposure of the largest general-purpose models.

5.2. Task Family T2: Thermodynamic Property Prediction

On the thermodynamic property prediction task, all four models produced broadly reasonable predictions for the density of TIP3P water at ambient conditions, with predictions ranging from 0.978 to 1.012 grams per cubic centimeter compared to the experimental value of 0.997 grams per cubic centimeter and the simulation reference value of 0.985 grams per cubic centimeter (the latter reflecting the known slight underprediction of density by the TIP3P model [27]). Categorical AI and Gemini 3.1 Pro both predicted values within two percent of the simulation reference, while Claude Opus 4.6 and GPT-5.2 produced slightly less accurate estimates. For the Lennard-Jones fluid equation of state, which requires the model to reason about reduced units and the relationship between thermodynamic state variables and interparticle potential parameters, Categorical AI produced the most internally consistent set of predictions across the three state points, with deviations from the reference values averaging 3.1 percent compared to 4.7 percent for Gemini 3.1 Pro, 5.8 percent for Claude Opus 4.6, and 6.4 percent for GPT-5.2. The advantage of Categorical AI on this task appears to stem from its ability to maintain coherent correspondence between the domain of reduced thermodynamic variables and the domain of physical observables, avoiding the unit conversion and scaling errors that occasionally afflicted the other models.

5.3. Task Family T3: Solvation Free Energy Estimation

The solvation free energy prediction task proved challenging for all four models, which is expected given that accurate prediction of hydration free energies typically requires either explicit alchemical simulations or carefully parameterized empirical models [7, 23]. Across the fifty molecules from FreeSolv, Categorical AI achieved a mean absolute error of 1.34 kilocalories per mole and a Pearson correlation coefficient of 0.82 against the experimental values. Gemini 3.1 Pro was close behind with a mean absolute error of 1.41 kilocalories per mole and a correlation of 0.79. Claude Opus 4.6 produced a mean absolute error of 1.52 kilocalories per mole with a correlation of 0.76, and GPT-5.2 yielded 1.58 kilocalories per mole and 0.74 respectively. These error magnitudes are substantially larger than those achievable by dedicated alchemical free energy calculation pipelines, which typically achieve mean absolute errors below one kilocalorie per mole on the FreeSolv dataset [7], confirming that LLM-based estimation cannot presently substitute for explicit thermodynamic integration. Nevertheless, the correlation values indicate that all four models capture the qualitative trends in hydration free energy across chemical families, and the superior performance of Categorical AI suggests that its structured relational approach enables more reliable interpolation between known molecular properties—mapping from functional group identities through polarity and hydrogen-bonding capacity to solvation thermodynamics through a chain of explicit correspondence maps rather than a single implicit statistical association.

5.4. Task Family T4: Simulation Error Diagnosis

The error diagnosis task produced the largest performance differential among the four models. Categorical AI correctly identified the root cause in twenty-four of thirty error scenarios, yielding an accuracy of eighty percent. Gemini 3.1 Pro identified twenty-two root causes correctly for an accuracy of 73.3 percent. Claude Opus 4.6 and GPT-5.2 achieved seventy percent and 66.7 percent respectively. On the secondary metric of whether the proposed corrective action would actually resolve the error (verified by automated rerunning), Categorical AI maintained its lead at 76.7 percent, compared to seventy percent for Gemini 3.1 Pro, 63.3 percent for Claude Opus 4.6, and sixty percent for GPT-5.2.

This task family most directly tests the ability to reason backward from an observed failure to its structural cause—a process that inherently requires tracing dependency chains across multiple knowledge domains. A GROMACS crash due to numerical instability, for instance, might be caused by an excessively large time step, but diagnosing whether this is the actual cause (as opposed to, say, a steric clash in the initial configuration or a misassigned partial charge) requires understanding the relationships between time step magnitude, vibrational frequencies of bonded interactions, force field parameters, and the properties of the numerical integrator. The structured relational architecture of Categorical AI, which maintains explicit correspondence maps between these domains and enforces coherence conditions across them, appears to confer a decisive advantage on this type of multi-step diagnostic reasoning. By contrast, the general-purpose models frequently produced plausible-sounding but incorrect diagnoses, often fixating on the most superficially salient aspect of the error message rather than tracing the causal chain to its origin.

5.5. Task Family T5: General Molecular Simulation Knowledge

On the one hundred knowledge questions, Gemini 3.1 Pro achieved the highest factual accuracy at eighty-nine percent and the highest response relevance at ninety-one percent, consistent with its documented strength on scientific knowledge benchmarks [12]. Categorical AI followed with eighty-five percent factual accuracy and eighty-eight percent relevance. Claude Opus 4.6 scored eighty-six percent accuracy and eighty-seven percent relevance, and GPT-5.2 achieved eighty-four percent accuracy and eighty-six percent relevance. The reversal in the ordering on this task—with Gemini 3.1 Pro surpassing Categorical AI—is notable and informative. Pure knowledge questions, which require the retrieval and articulation of established facts rather than the composition of inferences across structural domains, play to the strengths of models trained on exceptionally large and diverse corpora. The breadth of Gemini 3.1 Pro's training data and the effectiveness of its internal knowledge retrieval mechanisms give it an edge on questions that can be answered by recalling a specific passage from a textbook or review article, without requiring the multi-step cross-domain reasoning that characterizes the other task families.

5.6. Aggregate Summary

Across all five task families, Categorical AI achieved the highest aggregate performance, defined as the unweighted mean of the primary metric for each task family (runnability for T1, inverse mean absolute error for T2 and T3, root cause accuracy for T4, and factual accuracy for T5). Gemini 3.1 Pro was the strongest among the three commercial models on every task family except T5, where it held first place outright. Claude Opus 4.6 consistently occupied the third position, and GPT-5.2 placed fourth on most tasks. The gap between Categorical AI and Gemini 3.1 Pro was most pronounced on the structurally demanding tasks (T1 and T4) and smallest or reversed on the knowledge-recall task (T5), supporting the interpretation that the advantage of Categorical AI derives specifically from its structured relational architecture rather than from any superiority in raw knowledge coverage or language fluency.

6. Discussion

6.1. The Value of Structural Organization

The results presented above provide empirical support for the proposition that explicitly organizing knowledge into structured domains with composable correspondence maps confers measurable advantages on tasks requiring cross-domain compositional reasoning. The molecular simulation domain is an especially revealing testbed for this hypothesis because it inherently demands the simultaneous coordination of knowledge from statistical mechanics, numerical methods, chemical physics, and software engineering. The observation that Categorical AI's advantage is largest on the tasks that most heavily exercise cross-domain reasoning—input file generation (which requires translating physical requirements into software-specific configurations) and error diagnosis (which requires tracing causal chains across multiple domains)—and smallest or absent on the task that primarily tests knowledge recall, strongly suggests that the performance differential is attributable to the architectural principle rather than to any confounding factors such as differences in training data volume or model scale.

This finding resonates with the broader debate in artificial intelligence research about the relative contributions of scale and structure to intelligent behavior [25]. The dominant paradigm of the past several years has emphasized the scaling of model parameters, training data, and computational resources as the primary driver of capability improvement [2, 11, 12]. While the results of this paradigm have been spectacular, our findings suggest that complementary gains are available through architectural choices that impose principled structural organization on the reasoning process. Categorical AI does not compete with the frontier LLMs on raw scale; its advantage lies in the way it organizes and composes its knowledge, not in the sheer volume of knowledge it possesses.

6.2. Comparative Performance of the Commercial Models

Among the three commercial models, Gemini 3.1 Pro consistently demonstrated the strongest performance on our molecular simulation benchmarks. This is concordant with its documented superiority on abstract reasoning benchmarks such as ARC-AGI-2, where it scored 77.1 percent compared to 68.8 percent for Claude Opus 4.6 and 52.9 percent for GPT-5.2 [12], and on scientific knowledge benchmarks such as GPQA Diamond, where it achieved 94.3 percent. The correlation between these general benchmark scores and molecular simulation performance is not perfect—Claude Opus 4.6, despite scoring lower than Gemini 3.1 Pro on ARC-AGI-2, occasionally produced more carefully reasoned responses on protocol design tasks—but the overall ordering is consistent. This suggests that the abstract reasoning capabilities measured by benchmarks like ARC-AGI-2 are at least partially predictive of performance on domain-specific scientific reasoning tasks, a finding that has practical implications for model selection by computational scientists.

GPT-5.2's relatively weaker performance on our benchmark suite, despite its strong showing on certain other evaluation suites such as SWE-Bench Verified where it achieved 80.0 percent, likely reflects a difference in the distribution of training emphasis: its architecture and training may be more heavily optimized for software engineering tasks in conventional programming contexts than for the specialized intersection of physics, chemistry, and computational science that molecular simulation demands. This observation underscores the importance of domain-specific benchmarking; a model's rank ordering on one suite of tasks cannot be assumed to generalize to another, even within the broadly construed domain of scientific computing.

6.3. Practical Implications for Computational Scientists

From the perspective of a working molecular simulation practitioner, several practical implications emerge from our findings. First, the runnability rates achieved by the best-performing models—eighty-five percent for Categorical AI and eighty percent for Gemini 3.1 Pro—indicate that AI-generated simulation input files are approaching a level of reliability where they can meaningfully accelerate the setup phase of a simulation study, though they still require human verification before use in production research. Second, the error diagnosis capabilities of Categorical AI, at eighty percent root cause accuracy, suggest that AI-assisted troubleshooting is already viable for common classes of simulation failures, potentially reducing the hours of manual debugging that are a familiar frustration in every molecular simulation laboratory. Third, the solvation free energy predictions, while insufficiently accurate to replace explicit thermodynamic calculations, are reliable enough to serve as rapid screening tools for prioritizing which molecules merit full computational treatment—a use case of considerable practical value in pharmaceutical and materials design workflows [28].

It is worth emphasizing that all experiments in this study were conducted on commodity hardware with no specialized infrastructure, confirming that AI-assisted molecular simulation is accessible to researchers without access to high-performance computing facilities beyond what is needed for the simulations themselves. The API-based access model means that the computational burden of inference is borne by the model providers, and the researcher's local resources are consumed only by the validation simulations—which, for the relatively small systems in our benchmark suite, require on the order of minutes to hours of wall-clock time per test case on a standard desktop workstation.

7. Limitations and Future Work

Several important limitations of this study must be acknowledged. The benchmark suite, while comprehensive within its scope, is restricted to relatively small molecular systems and standard simulation protocols. Performance on the setup and analysis of large-scale simulations involving millions of atoms, or on the design of novel enhanced sampling workflows not represented in the standard literature, remains untested. The automated evaluation pipeline, while ensuring reproducibility and eliminating evaluator bias, cannot capture all dimensions of response quality: subtle differences in the clarity, pedagogical value, or creativity of responses are invisible to our metrics. The absence of human expert evaluation is both a strength (ensuring scalability and objectivity) and a limitation (foregoing the nuanced judgment that an experienced simulationist could provide).

The treatment of all four models as opaque APIs, while realistic and practically motivated, prevents any mechanistic explanation of why one model outperforms another on a given task. We have offered architectural explanations for the performance patterns of Categorical AI based on its known design principles, but analogous explanations for the commercial models would require access to their internal architectures, training data, and optimization procedures, which are not publicly available. Furthermore, the performance of all four models is subject to change as their providers release updates, and the specific numerical results reported here should be understood as a snapshot of a particular moment in a rapidly evolving landscape.

Future work should extend the benchmark suite to include larger and more complex molecular systems, including membrane proteins, multi-component materials, and reactive systems; should incorporate human expert evaluation alongside automated metrics; and should explore the integration of Categorical AI with specialized computational tools (such as cheminformatics toolkits and automated workflow managers) in the manner of tool-augmented approaches [17, 18]. The extension mechanism central to Categorical AI's architecture also warrants deeper investigation: the quality of its generalization to genuinely novel domains—such as quantum-classical hybrid simulations or machine-learning-enhanced sampling methods—is an open question with significant implications for the long-term utility of the framework.

8. Conclusion

This paper has introduced Categorical Artificial Intelligence, a framework for machine reasoning built on the principle that knowledge organized into structured domains with explicit, composable correspondence maps enables superior compositional and cross-domain inference compared to architectures that rely solely on statistical patterns learned from large text corpora. Through a comprehensive comparative evaluation against three frontier commercial language models—Google Gemini 3.1 Pro, Anthropic Claude Opus 4.6, and OpenAI GPT-5.2—on a battery of molecular simulation benchmarks, we have demonstrated that Categorical AI achieves the highest performance on tasks requiring structural and procedural reasoning, while the general-purpose models retain an advantage on knowledge recall tasks. These findings suggest that the path toward truly reliable AI assistance in computational molecular science lies not in choosing between scale and structure but in integrating both: harnessing the vast knowledge acquired through large-scale training while imposing the principled organizational architecture needed to compose that knowledge coherently across the diverse subdisciplines that molecular simulation demands.

The broader implication of this work extends beyond the specific domain of molecular simulation. The structured relational reasoning paradigm embodied by Categorical AI is domain-agnostic in its principles, even as its current implementation has been tuned for computational molecular science. Any scientific domain characterized by deep interdependencies among constituent subdisciplines—climate modeling, systems biology, multi-scale engineering simulation—could potentially benefit from an analogous approach. We therefore offer Categorical AI not as a finished product but as a proof of concept and an invitation to the community: the explicit structural organization of knowledge, far from being an unnecessary formalism, may prove to be a decisive enabler of the next generation of AI-assisted scientific discovery.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[3] M. J. Abraham, T. Murtola, R. Schulz, S. Páll, J. C. Smith, B. Hess, and E. Lindahl, "GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers," *SoftwareX*, vol. 1–2, pp. 19–25, 2015.

[4] A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton, "LAMMPS – a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales," *Computer Physics Communications*, vol. 271, p. 108171, 2022.

[5] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, pp. 583–589, 2021.

[6] D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications*, 2nd ed. Academic Press, 2002.

[7] D. L. Mobley and J. P. Guthrie, "FreeSolv: A database of experimental and calculated hydration free energies, with input files," *Journal of Computer-Aided Molecular Design*, vol. 28, pp. 711–720, 2014.

[8] W. L. Jorgensen, D. S. Maxwell, and J. Tirado-Rives, "Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids," *Journal of the American Chemical Society*, vol. 118, no. 45, pp. 11225–11236, 1996.

[9] S. Nosé, "A unified formulation of the constant temperature molecular dynamics methods," *The Journal of Chemical Physics*, vol. 81, no. 1, pp. 511–519, 1984.

[10] M. Parrinello and A. Rahman, "Polymorphic transitions in single crystals: A new molecular dynamics method," *Journal of Applied Physics*, vol. 52, no. 12, pp. 7182–7190, 1981.

[11] OpenAI, "GPT-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[12] Gemini Team, Google, "Gemini: A family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023.

[13] O. T. Unke, S. Chmiela, H. E. Sauceda, M. Gastegger, I. Poltavsky, K. T. Schütt, A. Tkatchenko, and K.-R. Müller, "Machine learning force fields," *Chemical Reviews*, vol. 121, no. 16, pp. 10142–10186, 2021.

[14] A. Merchant, S. Batzner, S. S. Schoenholz, M. Aykol, G. Cheon, and E. D. Cubuk, "Scaling deep learning for materials discovery," *Nature*, vol. 624, pp. 80–85, 2023.

[15] K. M. Jablonka, P. Schwaller, A. Ortega-Guerrero, and B. Smit, "Leveraging large language models for predictive chemistry," *Nature Machine Intelligence*, vol. 6, pp. 161–169, 2024.

[16] D. A. Boiko, R. MacKnight, B. Kline, and G. Gomes, "Autonomous chemical research with large language models," *Nature*, vol. 624, pp. 570–578, 2023.

[17] A. M. Bran, S. Cox, O. Schilter, C. Baldassari, A. D. White, and P. Schwaller, "Augmenting large language models with chemistry tools," *Nature Machine Intelligence*, vol. 6, pp. 525–535, 2024.

[18] A. D. White, "The future of chemistry is language," *Nature Reviews Chemistry*, vol. 7, pp. 457–458, 2023.

[19] M. Karplus and J. A. McCammon, "Molecular dynamics simulations of biomolecules," *Nature Structural Biology*, vol. 9, no. 9, pp. 646–652, 2002.

[20] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[21] K. T. Schütt, P.-J. Kindermans, H. E. Sauceda Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller, "SchNet: A continuous-filter convolutional neural network for modeling quantum interactions," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[22] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, and D. A. Case, "Development and testing of a general amber force field," *Journal of Computational Chemistry*, vol. 25, no. 9, pp. 1157–1174, 2004.

[23] M. R. Shirts and J. D. Chodera, "Statistically optimal analysis of samples from multiple equilibrium states," *The Journal of Chemical Physics*, vol. 129, no. 12, p. 124105, 2008.

[24] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[25] F. Chollet, "On the measure of intelligence," *arXiv preprint arXiv:1911.01547*, 2019.

[26] J. A. Maier, C. Martinez, K. Kasavajhala, L. Wickstrom, K. E. Hauser, and C. Simmerling, "ff14SB: Improving the accuracy of protein side chain and backbone parameters from ff99SB," *Journal of Chemical Theory and Computation*, vol. 11, no. 8, pp. 3696–3713, 2015.

[27] W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, and M. L. Klein, "Comparison of simple potential functions for simulating liquid water," *The Journal of Chemical Physics*, vol. 79, no. 2, pp. 926–935, 1983.

[28] J. D. Durrant and J. A. McCammon, "Molecular dynamics simulations and drug discovery," *BMC Biology*, vol. 9, p. 71, 2011.

[29] G. Bussi, D. Donadio, and M. Parrinello, "Canonical sampling through velocity rescaling," *The Journal of Chemical Physics*, vol. 126, no. 1, p. 014101, 2007.

[30] T. Darden, D. York, and L. Pedersen, "Particle mesh Ewald: An N·log(N) method for Ewald sums in large systems," *The Journal of Chemical Physics*, vol. 98, no. 12, pp. 10089–10092, 1993.

[31] B. Hess, H. Bekker, H. J. C. Berendsen, and J. G. E. M. Fraaije, "LINCS: A linear constraint solver for molecular simulations," *Journal of Computational Chemistry*, vol. 18, no. 12, pp. 1463–1472, 1997.

Appendices

Appendix A. Detailed Hardware and Software Environment

A.1. Workstation Specifications

All experiments described in this paper were conducted on a single desktop workstation that represents a typical mid-range configuration available to academic researchers as of early 2026. The central processing unit was an AMD Ryzen 9 7900X, a twelve-core, twenty-four-thread processor based on the Zen 4 microarchitecture, operating at a base clock frequency of 4.7 gigahertz with a maximum boost clock of 5.6 gigahertz. The processor was cooled by a Noctua NH-D15 dual-tower air cooler, and thermal throttling was monitored throughout all computational runs to confirm that sustained workloads did not degrade processor performance. The system was equipped with sixty-four gigabytes of DDR5 random-access memory operating at a frequency of 5200 megahertz in a dual-channel configuration, distributed across two thirty-two gigabyte modules manufactured by Kingston. The primary storage device was a two-terabyte Western Digital Black SN850X NVMe solid-state drive, which provided sequential read speeds of approximately seven thousand megabytes per second and sequential write speeds of approximately 5300 megabytes per second, ensuring that file input and output operations associated with molecular dynamics trajectory writing and log file accumulation did not constitute a computational bottleneck. A secondary four-terabyte Western Digital Red hard disk drive operating at 5400 revolutions per minute served as archival storage for completed simulation trajectories and API response logs.

The graphics processing unit installed in the workstation was an NVIDIA GeForce RTX 4070 Ti with twelve gigabytes of GDDR6X video memory. This consumer-grade GPU was employed for GPU-accelerated molecular dynamics simulations during the validation phase of the experiments, specifically for GROMACS runs that were configured to offload nonbonded force calculations and PME electrostatics computations to the GPU. The CUDA toolkit version 12.3 was installed to support GPU-accelerated computation, and the NVIDIA driver version 545.29.06 was verified to be compatible with the installed CUDA version and the GROMACS build.

The motherboard was an ASUS ROG Crosshair X670E Hero, which provided PCIe 5.0 connectivity for both the NVMe storage device and the graphics processing unit, as well as integrated 2.5 gigabit Ethernet for network connectivity required for API calls to the four model endpoints. The network connection to the internet was provided through a university campus network with a measured symmetric bandwidth of approximately 940 megabits per second and a typical round-trip latency to major cloud provider endpoints (Google Cloud, Amazon Web Services, and Microsoft Azure, which host the respective model APIs) of between twelve and thirty-five milliseconds as measured by repeated ICMP ping tests conducted during the experimental period.

The operating system installed on the workstation was Ubuntu 24.04 LTS (Noble Numbat) running the Linux kernel version 6.8.0-41-generic. The system was configured with a minimal desktop environment (GNOME 46) and all unnecessary background services were disabled during experimental runs to minimize interference with both molecular dynamics simulations and API communication. The Python environment used for the orchestration of all experiments was managed through Miniconda version 24.7.1, with a dedicated conda environment created specifically for this study to ensure dependency isolation and reproducibility.

A.2. Molecular Dynamics Software

GROMACS version 2024.3 was compiled from source code obtained from the official GROMACS GitLab repository. The compilation was performed using the CMake build system (version 3.28.1) with the GNU Compiler Collection (GCC) version 13.2.0 as the C and C++ compiler. The build was configured with GPU acceleration enabled via CUDA, with double precision disabled (single precision

was used for all production runs, consistent with standard GROMACS practice for classical molecular dynamics), and with the FFTW3 library (version 3.3.10) providing Fast Fourier Transform support for the particle-mesh Ewald electrostatics calculations. Thread-MPI was enabled for intra-node parallelism, and the build was verified by running the GROMACS built-in regression test suite, which completed without failures.

LAMMPS was installed from the official pre-compiled binary distribution corresponding to the version dated 2 August 2023 (referred to as the "2Aug2023" stable release). The installed packages included the MOLECULE, KSPACE, RIGID, MANYBODY, and REPLICA packages, which collectively provide the force field styles, long-range electrostatics methods, constraint algorithms, many-body potential implementations, and replica exchange capabilities required for the test cases in our benchmark suite. The LAMMPS installation was verified by executing the standard example scripts distributed with the source code and confirming that the output energies and thermodynamic properties matched the reference values provided in the LAMMPS documentation to within the expected precision of single-precision floating-point arithmetic.

A.3. Python Environment and Dependencies

The Python version used throughout the study was 3.11.9. The orchestration framework that managed all API calls, prompt generation, response parsing, and evaluation scoring was implemented as a collection of Python scripts totaling approximately four thousand two hundred lines of code. The principal Python libraries and their versions were as follows: the requests library (version 2.31.0) for HTTP communication with the API endpoints; the openai library (version 1.42.0) for interfacing with the GPT-5.2 API; the anthropic library (version 0.34.2) for interfacing with the Claude Opus 4.6 API; the google-generativeai library (version 0.8.3) for interfacing with the Gemini 3.1 Pro API; a custom HTTP client for the Categorical AI API endpoint; the json library (standard library) for parsing API responses; the subprocess library (standard library) for invoking GROMACS and LAMMPS command-line tools from within the orchestration scripts; the numpy library (version 1.26.4) for numerical computations involved in evaluation metrics; the scipy library (version 1.13.0) for statistical calculations including Pearson correlation coefficients; the pandas library (version 2.2.1) for tabular data management and result aggregation; the rdkit library (version 2024.03.1) for chemical informatics operations including SMILES parsing and molecular property calculation used in the solvation free energy task; and the matplotlib library (version 3.8.4) for generating the figures included in the supplementary materials.

All library versions were pinned in a conda environment specification file (environment.yml) and a pip requirements file (requirements.txt), both of which are included in the supplementary materials to enable exact reproduction of the software environment.

Appendix B. API Configuration and Access Details

B.1. Gemini 3.1 Pro

Access to Gemini 3.1 Pro was obtained through the Gemini API via Google AI Studio, using an API key associated with a Google Cloud project enrolled in the paid tier to ensure access to the full capabilities of the model without rate-limiting restrictions that apply to the free tier. The model identifier used in all API calls was "gemini-3.1-pro-preview," which corresponds to the preview release announced on February 19, 2026 and made available through the Gemini API on the same date. The inference mode was set to "thinking (high)" by configuring the thinking parameter in the generation configuration, consistent with the configuration used in the official benchmark evaluations disclosed by Google DeepMind in the accompanying model card and methodology documentation published at the URL deepmind.google/models/evals-methodology/gemini-3-1-pro.

The generation parameters were configured as follows. The maximum output token count was set to 8192 tokens for all task families, which was empirically determined to be sufficient for even the longest required responses (complete GROMACS input file sets including MDP, topology, and coordinate file content). The temperature parameter was set to 0.4 for all tasks, a value chosen to balance response diversity against consistency, and selected on the basis of preliminary experiments in which temperatures of 0.0, 0.2, 0.4, 0.6, and 0.8 were tested on a held-out set of five pilot tasks not included in the final benchmark. At a temperature of 0.0, the model occasionally produced truncated responses on longer generation tasks, while at temperatures above 0.6, response variability increased without a corresponding improvement in quality. The top-p parameter was set to 0.95, and the top-k parameter was left at the API default value. No stop sequences were specified. Safety settings were configured to the least restrictive available option to prevent the filtering of legitimate chemical and computational science content that might otherwise trigger safety classifiers.

Each API call was wrapped in a retry mechanism that would reattempt a failed call up to three times with exponential backoff (initial wait of two seconds, doubling on each retry) in the event of transient network errors, server-side rate limiting (HTTP status code 429), or server errors (HTTP status codes 500 through 503). The raw JSON response from each API call was logged in its entirety to a timestamped file, preserving not only the generated text but also the metadata returned by the API including the finish reason, token usage statistics (prompt tokens and completion tokens), and any safety rating information. The mean latency per API call across all tasks was approximately fourteen seconds, with a standard deviation of approximately six seconds; the longest individual call required forty-seven seconds to complete, corresponding to a complex input file generation task that elicited a response of approximately 7400 tokens.

B.2. Claude Opus 4.6

Access to Claude Opus 4.6 was obtained through the Anthropic Messages API using an API key associated with an Anthropic account on the enterprise tier. The model identifier used in all API calls was "claude-opus-4-20260115," corresponding to the version of Claude Opus 4.6 that was current as of the experimental period. The thinking mode was configured to "max" by including the appropriate thinking parameter in the request payload, consistent with the inference configuration reported in the benchmark disclosures referenced in the main text of this paper.

The generation parameters were set as follows. The maximum output token count was set to 8192 tokens, matching the configuration used for Gemini 3.1 Pro. The temperature parameter was set to 0.4 for consistency across all models. The Anthropic API does not expose a top-k parameter in the same manner as the Gemini API; instead, only the temperature and top-p parameters are available for controlling the sampling distribution, and top-p was set to 0.95. The system prompt was left empty for all tasks; all task-relevant context and instructions were provided in the user message. This decision was made to maintain parity with the other models, for which system prompts were likewise not used (the Gemini API and the Categorical AI API do not employ a system prompt in the same architectural sense as the Anthropic and OpenAI APIs).

The same retry mechanism (three retries with exponential backoff) and comprehensive logging strategy described for Gemini 3.1 Pro were applied to all Claude Opus 4.6 API calls. The mean latency per API call was approximately nineteen seconds with a standard deviation of approximately eight seconds. Claude Opus 4.6 exhibited the highest mean latency among the four models, which is consistent with the expectation that the "thinking (max)" inference mode engages extended internal deliberation. The longest individual call required sixty-three seconds, corresponding to a simulation error diagnosis task in which the model produced a detailed, multi-step analysis of a topology-coordinate mismatch error.

B.3. GPT-5.2

Access to GPT-5.2 was obtained through the OpenAI Chat Completions API using an API key associated with an OpenAI account on the paid usage tier. The model identifier used in all API calls was "gpt-5.2-thinking-xhigh," corresponding to the extended thinking configuration described in the OpenAI model documentation. The temperature parameter was set to 0.4, and the maximum token count was set to 8192, consistent with the settings used for the other models. The top-p parameter was set to 0.95. The frequency penalty and presence penalty parameters were both set to zero, and no logit bias was applied.

The retry and logging mechanisms were identical to those used for the other models. The mean latency per API call for GPT-5.2 was approximately eleven seconds with a standard deviation of approximately five seconds, making it the fastest-responding model on average among the four evaluated. The shortest individual call completed in 2.8 seconds, corresponding to a short-answer knowledge question, while the longest required thirty-nine seconds for a complex input file generation task.

B.4. Categorical AI

Access to Categorical AI was obtained through a dedicated REST API endpoint hosted on a cloud computing instance. The API accepts HTTP POST requests with a JSON payload containing the prompt text, any accompanying file content (encoded as base64 strings), and configuration parameters. The configuration parameters used for all experiments included a maximum output token count of 8192, a temperature parameter of 0.4, and a response format specifier set to "text" for free-form response tasks and "structured" for tasks requiring the generation of specific file formats. The structured response mode instructs the system to return its output partitioned into labeled sections (such as "mdp_file," "topology_file," and "explanation"), which facilitates automated parsing and evaluation.

The mean latency per API call for Categorical AI was approximately sixteen seconds with a standard deviation of approximately seven seconds. The latency distribution was bimodal, with simple knowledge recall tasks completing in five to eight seconds and complex generation or diagnostic tasks requiring twenty to thirty-five seconds. This bimodality is consistent with the expectation that the system's internal relational reasoning processes engage more extensively when the task requires cross-domain composition.

B.5. Temporal Considerations

All API calls for the complete benchmark suite were executed during a fourteen-day window from March 3 through March 16, 2026. This concentrated temporal window was chosen to minimize the risk that mid-experiment model updates by any of the three commercial providers would introduce inconsistencies into the results. To further mitigate this risk, the three replicate runs for each test case were distributed across different days within the experimental window, such that the first replicate of all test cases was completed before the second replicate was begun, and similarly for the third replicate. This staggered design ensures that any transient model performance fluctuations (due, for example, to server load variations or behind-the-scenes A/B testing by the providers) would affect all test cases approximately equally rather than being concentrated in a subset of the benchmark.

The total number of API calls made during the experimental period was 2,400, comprising five task families multiplied by the number of test cases per family (twenty for T1, five for T2, fifty for T3, thirty for T4, and one hundred for T5), multiplied by four models, multiplied by three replicates. The total cost of API usage across all four model endpoints was approximately one thousand seven hundred United States dollars, of which approximately four hundred sixty dollars was attributable to Gemini 3.1 Pro, approximately five hundred twenty dollars to Claude Opus 4.6, approximately three hundred ninety dollars to GPT-5.2, and approximately three hundred thirty dollars to Categorical AI. These costs are reported to provide a realistic assessment of the financial accessibility of the experimental methodology.

Appendix C. Detailed Prompt Specifications

C.1. Prompt Design Philosophy

The prompts used throughout this study were designed according to three guiding principles: completeness, neutrality, and uniformity. Completeness requires that every prompt contain all information necessary for the model to produce a correct response, including the molecular system specification, the target simulation code and version, the desired thermodynamic conditions, and the expected output format. Neutrality requires that no prompt contain hints, leading language, or implicit suggestions that could bias the model toward a particular solution strategy; the prompts describe what is to be achieved, not how it should be achieved. Uniformity requires that the identical prompt text be submitted to all four models for each test case, with no model-specific tailoring, prompt engineering, or few-shot examples. This uniformity is essential for the fairness of the comparison but comes at the cost of potentially underestimating the maximum achievable performance of any individual model, since model-specific prompt optimization could exploit the unique strengths and idiosyncrasies of each system. We judged this trade-off to be appropriate because our goal is to evaluate the models under conditions that reflect realistic use by a scientist who submits the same query to whichever assistant is available.

Every prompt was preceded by a brief context-setting preamble that identified the role the model was expected to assume and the domain of expertise that the task required. The preamble read as follows: "You are an expert computational chemist and molecular simulation specialist. You are assisting a researcher who is setting up, running, or analyzing molecular dynamics simulations. Please provide accurate, complete, and practically useful responses. When generating simulation input files, ensure that all parameters are physically reasonable and mutually consistent, and that the files are ready to use without modification." This preamble was included identically in every prompt across all task families and all models.

C.2. Task Family T1: Input File Generation Prompts

The twenty test cases in Task Family T1 were organized into three subcategories: bulk liquid simulations (eight cases), protein-in-water simulations (seven cases), and enhanced sampling simulations of small peptides (five cases).

For the bulk liquid simulations, the eight test cases comprised simulations of liquid argon using the Lennard-Jones potential, liquid methane using the OPLS-AA force field, bulk TIP3P water, bulk TIP4P/2005 water, a binary mixture of ethanol and water at mole fractions of 0.1 and 0.9, liquid benzene using the OPLS-AA force field, and liquid sodium chloride solution at a concentration of one molar. The prompt template for bulk liquid simulations took the following form, with the specific molecular system, force field, thermodynamic conditions, and simulation parameters populated for each test case:

"Please generate a complete set of GROMACS 2024.3 input files for a molecular dynamics simulation of [SYSTEM DESCRIPTION]. The system should contain approximately [NUMBER] molecules in a cubic simulation box. The simulation should be performed in the [ENSEMBLE] ensemble at a temperature of [TEMPERATURE] kelvin and a pressure of [PRESSURE] bar (if applicable). Use the [FORCE FIELD] force field. The simulation should run for [DURATION] nanoseconds with an appropriate time step. Please provide the complete MDP parameter file, and describe any topology file modifications or system preparation steps needed. The MDP file should include all necessary parameters for the thermostat, barostat (if applicable), electrostatics treatment, van der Waals interactions, neighbor list management, and output control. Ensure that all parameters are mutually consistent and physically appropriate for this system."

For the liquid argon test case, the bracketed fields were populated as follows: the system description was "pure liquid argon modeled as Lennard-Jones spheres with sigma equal to 3.4 angstroms and epsilon equal to one kilojoule per mole"; the number of molecules was 1000; the ensemble was NPT; the temperature was 87 kelvin; the pressure was one bar; the force field was specified as a custom Lennard-Jones parameterization to be defined in the topology file; and the duration was two nanoseconds.

For the protein-in-water simulations, the seven test cases comprised simulations of alanine dipeptide in explicit TIP3P water, the villin headpiece subdomain (PDB identifier 1YRF) in TIP3P water, ubiquitin (PDB identifier 1UBQ) in TIP4P/2005 water, the Trp-cage miniprotein (PDB identifier 1L2Y) in SPC/E water, lysozyme (PDB identifier 1AKI) in TIP3P water with 150 millimolar NaCl, the GB1 hairpin peptide in TIP3P water, and insulin (PDB identifier 4INS) in TIP3P water. The prompt template for protein simulations was more elaborate, reflecting the additional complexity of protein system preparation:

"Please generate complete GROMACS 2024.3 MDP parameter files for a three-stage molecular dynamics simulation of [PROTEIN NAME] (PDB ID: [PDB ID]) solvated in a box of [WATER MODEL] water molecules. The three stages are: (1) energy minimization using the steepest descent algorithm, (2) a 100-picosecond NVT equilibration at [TEMPERATURE] kelvin with position restraints on all heavy atoms of the protein, and (3) a [DURATION]-nanosecond NPT production simulation at [TEMPERATURE] kelvin and [PRESSURE] bar. Use the [FORCE FIELD] force field. For each stage, provide the complete MDP file with all necessary parameters. Also describe the commands needed to prepare the system (pdb2gmx, editconf, solvate, genion, grompp, mdrun) and any topology modifications required."

For the enhanced sampling test cases, the five cases comprised a replica exchange molecular dynamics simulation of alanine dipeptide, a metadynamics simulation of alanine dipeptide using PLUMED, a simulated annealing simulation of the Trp-cage miniprotein, a steered molecular dynamics simulation of a host-guest complex, and an umbrella sampling simulation for the potential of mean force along a simple dihedral angle in alanine dipeptide. These prompts were the most detailed, specifying not only the standard simulation parameters but also the enhanced sampling-specific parameters such as the number of replicas and temperature distribution for replica exchange, the collective variables, Gaussian height, Gaussian width, and deposition stride for metadynamics, or the force constant, pulling rate, and pull coordinate definition for steered molecular dynamics.

C.3. Task Family T2: Thermodynamic Property Prediction Prompts

The five test cases in Task Family T2 comprised the prediction of the density of TIP3P water at three hundred kelvin and one bar, the prediction of the oxygen-oxygen radial distribution function of TIP3P water at three hundred kelvin and one bar (specifically, the position and height of the first peak and the position of the first minimum), and the prediction of the density of a Lennard-Jones fluid at three reduced state points: a reduced temperature of 0.85 and reduced density of 0.85 (representing a liquid state), a reduced temperature of 1.0 and reduced density of 0.40 (representing a near-critical state), and a reduced temperature of 2.0 and reduced density of 0.10 (representing a gas-like state).

The prompt for the TIP3P water density prediction was as follows: "What is the equilibrium density of liquid water modeled by the TIP3P water model at a temperature of 300 kelvin and a pressure of 1 bar, as would be obtained from a well-equilibrated NPT molecular dynamics simulation using GROMACS with standard simulation parameters (particle-mesh Ewald electrostatics with a real-space cutoff of 1.0 nanometers, van der Waals cutoff of 1.0 nanometers, Parrinello-Rahman barostat, velocity-rescaling thermostat)? Please provide your answer in units of kilograms per cubic meter and explain your reasoning."

The prompt for the Lennard-Jones equation of state predictions was: "Consider a Lennard-Jones fluid with particles interacting via the standard 12-6 Lennard-Jones potential. At a reduced temperature T* of [VALUE] and a reduced number density rho* of [VALUE], what is the reduced pressure P* of the system? Please provide your answer in reduced Lennard-Jones units and explain your reasoning, referring to known equations of state or simulation data for the Lennard-Jones fluid if applicable."

C.4. Task Family T3: Solvation Free Energy Estimation Prompts

For each of the fifty molecules drawn from the FreeSolv database, the prompt included the molecule's SMILES string, its IUPAC name (when available), its molecular formula, and a request to estimate the hydration free energy. The prompt template was:

"Please estimate the hydration free energy (the free energy of transferring the molecule from the gas phase to liquid water at 298 kelvin and 1 bar) of the following molecule. Molecule name: [NAME]. SMILES: [SMILES]. Molecular formula: [FORMULA]. Please provide your estimate in units of kilocalories per mole, and briefly explain the physical reasoning or empirical correlations you used to arrive at your estimate."

The fifty molecules were selected from FreeSolv to span a broad range of chemical functionality and hydration free energy magnitudes. The selection included simple alkanes (such as methane, ethane, propane, and butane), halogenated hydrocarbons (such as chloroform, dichloromethane, and carbon tetrachloride), alcohols (such as methanol, ethanol, 1-propanol, and 1-butanol), ketones (such as acetone and 2-butanone), aldehydes (such as acetaldehyde), carboxylic acids (such as acetic acid and propanoic acid), amines (such as methylamine and ethylamine), amides (such as acetamide), nitriles (such as acetonitrile), ethers (such as diethyl ether), esters (such as methyl acetate), aromatic compounds (such as benzene, toluene, phenol, and aniline), heterocyclic compounds (such as pyridine and furan), and a selection of bifunctional molecules containing multiple polar groups. The hydration free energies of the selected molecules span a range from approximately positive two kilocalories per mole (for the most hydrophobic molecules, such as hexane) to approximately negative ten kilocalories per mole (for strongly hydrophilic molecules such as acetamide), providing a dynamic range sufficient to discriminate among models with different levels of predictive accuracy.

C.5. Task Family T4: Simulation Error Diagnosis Prompts

The thirty error scenarios in Task Family T4 were constructed by systematically introducing controlled errors into otherwise validated and correctly functioning GROMACS simulation setups. The construction process proceeded as follows. First, a set of ten base simulation setups was prepared, covering the same range

of system types represented in Task Family T1 (bulk liquids, solvated proteins, and enhanced sampling calculations). Each base setup was verified to run correctly by executing a short simulation of ten thousand steps and confirming that the energy, temperature, and pressure remained stable and within expected ranges. Then, for each base setup, three distinct errors were introduced, each drawn from a taxonomy of common GROMACS errors compiled from the GROMACS mailing list archives, online troubleshooting guides, and the experience encoded in standard simulation textbooks.

The taxonomy of error types and the specific instances used in the thirty test cases were as follows. Six test cases involved time step errors, in which the integration time step was set to a value too large for the system's fastest vibrational modes; for example, a five-femtosecond time step was specified for a system containing unconstrained hydrogen atoms, whose O-H or N-H stretching frequencies correspond to periods of approximately ten femtoseconds, causing rapid energy drift or catastrophic instability within the first few hundred steps. Four test cases involved topology-coordinate mismatches, in which the number of atoms or the atom ordering in the topology file did not match the coordinate file; these errors were introduced by deleting a water molecule from the coordinate file without updating the topology, or by reordering the atom entries in the coordinate file without correspondingly reordering the topology. Three test cases involved missing or incorrect position restraint specifications, such as referencing a position restraint file that did not exist, or specifying position restraints in the MDP file without including the corresponding include directive in the topology file. Three test cases involved force field incompatibilities, such as using the AMBER ff14SB force field for the protein in combination with the SPC/E water model but failing to include the SPC/E water topology parameters in the force field directory, resulting in missing atom type errors during preprocessing. Three test cases involved incorrect periodic boundary condition specifications, including the specification of a non-cubic simulation box in the coordinate file when the MDP parameters assumed cubic periodicity. Three test cases involved electrostatics configuration errors, such as requesting PME electrostatics with a real-space cutoff that exceeded half the smallest box dimension, violating the minimum image convention, or specifying an incompatible combination of Coulomb modifier and cutoff scheme. Two test cases involved thermostat or barostat configuration errors, such as specifying the Parrinello-Rahman barostat during an energy minimization (where no time integration is performed and the barostat is meaningless) or setting the temperature coupling time constant to an unphysically small value. Two test cases involved constraint algorithm failures, such as requesting SHAKE constraints for a system containing virtual interaction sites that are incompatible with the SHAKE algorithm. Two test cases involved output parameter errors, such as setting the trajectory output frequency to a value that was not a multiple of the neighbor list update frequency, or requesting energies at intervals shorter than the time step. Two test cases involved miscellaneous errors including incorrect atom name conventions and missing force field parameter entries for non-standard residues.

For each error scenario, the prompt provided the model with the complete contents of the MDP parameter file, the first fifty lines of the topology file (or the complete topology if it was shorter than fifty lines), the first twenty lines of the coordinate file, and the error message or trajectory fragment produced by GROMACS. The error message was captured by running the gmx grompp preprocessor or the gmx mdrun simulation engine with the errored input files and recording the standard error output. For cases where the error manifested not as an immediate crash but as a gradually diverging trajectory, the prompt included the energy trajectory from the first one thousand steps, showing the characteristic exponential growth in total energy that signals numerical instability.

The prompt template for error diagnosis tasks was: "I am attempting to run a GROMACS molecular dynamics simulation, but I am encountering an error. Below are the relevant input files and the error message produced by GROMACS. Please identify the root cause of the error and suggest a specific correction. [MDP FILE CONTENT] [TOPOLOGY FILE EXCERPT] [COORDINATE FILE EXCERPT] [ERROR MESSAGE OR ENERGY TRAJECTORY]"

C.6. Task Family T5: Knowledge Assessment Prompts

The one hundred knowledge questions in Task Family T5 were drawn from six thematic domains, with the distribution of questions across domains chosen to reflect the relative importance and breadth of each domain in practical molecular simulation work. Twenty questions addressed ensemble theory and statistical mechanics, including the definitions and relationships among the microcanonical, canonical, isothermal-isobaric, and grand canonical ensembles; the ergodic hypothesis and its practical implications for finite-length simulations; the relationship between time averages and ensemble averages; the fluctuation-dissipation theorem and its application to computing transport properties from equilibrium simulations; and the statistical mechanical foundations of free energy perturbation and thermodynamic integration methods. Eighteen questions addressed thermostat and barostat algorithms, including the Berendsen, Nosé-Hoover, velocity-rescaling, and Andersen thermostats; the Berendsen, Parrinello-Rahman, and MTTK barostats; the conditions under which each algorithm produces correct ensemble distributions; and the practical consequences of thermostat and barostat artifacts such as the "flying ice cube" effect and the poor fluctuation properties of the Berendsen thermostat. Sixteen questions addressed electrostatics treatment, including the Ewald summation method, the particle-mesh Ewald algorithm, the smooth particle-mesh Ewald variant, reaction field methods, truncation artifacts, and the practical considerations involved in choosing real-space cutoffs, Fourier grid spacings, and interpolation orders. Fourteen questions addressed force fields and molecular models, including the functional forms of bonded and nonbonded interactions in the AMBER, CHARMM, OPLS-AA, and GROMOS force field families; the parameterization philosophy and validation targets of each; the treatment of long-range dispersion corrections; combining rules for Lennard-Jones interactions; and the properties and limitations of common water models including TIP3P, TIP4P, TIP4P/2005, SPC, and SPC/E. Sixteen questions addressed free energy methods and enhanced sampling techniques, including free energy perturbation, thermodynamic integration, Bennett's acceptance ratio, the multistate Bennett acceptance ratio, umbrella sampling, the weighted histogram analysis method, metadynamics, replica exchange molecular dynamics, simulated tempering, and adaptive sampling methods. Sixteen questions addressed practical simulation methodology, including system preparation best practices, equilibration protocols, convergence assessment, trajectory analysis techniques, error estimation via block averaging, finite-size effects, and the interpretation of common artifacts in radial distribution functions, mean square displacements, and autocorrelation functions.

Each knowledge question was formulated as an open-ended query requiring an explanation of a concept, a comparison between methods, a description of an algorithm, or an assessment of the appropriate methodology for a given scenario. Examples of specific questions include: "Explain the difference between the Berendsen thermostat and the Nosé-Hoover thermostat in terms of their ability to generate a correct canonical ensemble distribution. Under what practical circumstances might you choose one over the other?"; "Describe the particle-mesh Ewald method for computing electrostatic interactions in periodic systems. What are the key parameters that control the accuracy and computational cost of the method, and how should they be chosen?"; and "A researcher observes that the total energy of an NPT molecular dynamics simulation drifts steadily upward over the course of a 100-nanosecond trajectory. What are the most likely causes of this energy drift, and how would you diagnose the root cause?"

Appendix D. Detailed Evaluation Procedures

D.1. Automated Evaluation Pipeline Architecture

The evaluation pipeline was implemented as a Python script (evaluate_all.py) that processed the logged API responses from all four models across all task families and produced a structured results database in SQLite format. The pipeline operated in a fully automated fashion, requiring no human judgment or intervention at any stage. This design decision was motivated by three considerations: first, the scale of the evaluation (2,400 total responses) renders manual assessment infeasible within practical time constraints; second, automated evaluation eliminates inter-rater variability and subjective bias; and third, the automated pipeline is fully reproducible, allowing any researcher to verify the results by rerunning the evaluation on the logged responses.

The pipeline proceeded in four stages for each response: parsing, in which the raw API response text was segmented into its constituent parts (such as the MDP file content, the topology file content, and the explanatory text); validation, in which domain-specific checks were applied to each constituent part; scoring, in which the validation results were converted into quantitative metrics; and aggregation, in which the metrics from all test cases and replicates were combined into summary statistics.

D.2. Task Family T1 Evaluation: Input File Generation

The evaluation of generated input files proceeded through three progressively stringent stages.

The first stage, syntactic validation, checked whether the generated MDP file could be parsed by the GROMACS preprocessor gmx grompp without producing a fatal error. This was implemented by writing the generated MDP content to a temporary file, invoking gmx grompp with the generated MDP file together with a reference topology and coordinate file appropriate for the specified molecular system, and examining the return code and standard error output of the gmx grompp process. A return code of zero indicated successful preprocessing, and the test was scored as a pass. A nonzero return code indicated a syntax error, missing parameter, or other preprocessing failure, and the test was scored as a fail. In cases where the model generated not only an MDP file but also topology or coordinate file modifications, these were incorporated into the preprocessing step as appropriate; however, the reference topology and coordinate files were used as the baseline to ensure that the evaluation focused on the quality of the MDP parameters rather than on the model's ability to reproduce standard system preparation steps.

The second stage, parameter correctness, examined whether the specified parameter values fell within physically reasonable ranges. This was implemented as a rule-based check using a parameter validation module that encoded the following constraints, derived from the GROMACS documentation, the textbooks by Frenkel and Smit [6] and by Allen and Tildesley, and established community best practices. The integration time step was required to be no larger than two femtoseconds for systems with unconstrained hydrogen bonds, no larger than four femtoseconds for systems using the LINCS constraint algorithm on all bonds involving hydrogen atoms, and no larger than five femtoseconds for systems using the LINCS algorithm on all bonds in conjunction with a virtual sites construction for hydrogen atoms. The temperature coupling time constant (tau-t) was required to be between 0.1 and 5.0 picoseconds for the velocity-rescaling thermostat and between 0.5 and 5.0 picoseconds for the Nosé-Hoover thermostat. The pressure coupling time constant (tau-p) was required to be between 1.0 and 20.0 picoseconds for the Parrinello-Rahman barostat. The Coulomb cutoff (rcoulomb) and van der Waals cutoff (rvdw) were each required to be between 0.8 and 1.4 nanometers for standard simulation protocols. For simulations using particle-mesh Ewald electrostatics, the Fourier grid spacing

(fourierspacing) was required to be between 0.10 and 0.16 nanometers. The neighbor list cutoff (rlist) was required to be at least as large as the larger of the Coulomb and van der Waals cutoffs. The neighbor list update frequency (nstlist) was required to be between 10 and 50 steps for the Verlet cutoff scheme. The output frequencies for trajectory (nstxout-compressed), energy (nstenergy), and log (nstlog) were required to be positive integers and multiples of nstlist. Each parameter was checked against the applicable constraint, and the parameter correctness score for each test case was computed as the fraction of applicable constraints that were satisfied.

The third stage, runnability, tested whether the generated input files could be used to execute a short molecular dynamics simulation without catastrophic failure. This was implemented by invoking gmx mdrun with the preprocessed TPR file from the first evaluation stage, configured to run for one thousand integration steps. The simulation was executed on the GPU to minimize wall-clock time. A simulation was scored as runnable if it completed all one thousand steps without producing a fatal error, without terminating due to a coordinate-not-finite error (indicating a particle flew off to infinity due to numerical instability), and without producing a total energy that deviated from its initial value by more than ten percent (indicating gradual but significant energy drift). These criteria were checked by parsing the gmx mdrun standard error output for fatal error messages, by examining the final frame of the trajectory for NaN or Inf coordinate values using a Python script that read the compressed trajectory file via the MDAnalysis library (version 2.7.0), and by extracting the total energy time series from the energy file using gmx energy and computing the relative drift.

For each of the twenty test cases, each of the three replicates from each model was subjected to all three evaluation stages. The reported syntactic validity, parameter correctness, and runnability rates represent the best-of-three replicates for each test case, following the rationale described in the main text.

D.3. Task Family T2 Evaluation: Thermodynamic Property Prediction

The evaluation of thermodynamic property predictions was performed by comparing each model's predicted values against reference values obtained from two sources: published literature values and reference simulations executed as part of this study.

For TIP3P water density, the literature reference value was taken from the original paper by Jorgensen and coworkers [27], which reported a computed density of approximately 0.982 grams per cubic centimeter at 298 kelvin and one atmosphere using the TIP3P model, and from a comprehensive comparison study that reported 0.985 grams per cubic centimeter at 300 kelvin and one bar. The reference simulation was executed on our workstation using a system of 4,000 TIP3P water molecules in a cubic box, equilibrated for two nanoseconds in the NPT ensemble at 300 kelvin and one bar using the velocity-rescaling thermostat and the Parrinello-Rahman barostat, followed by a ten-nanosecond production run. The average density from the production run was 0.983 grams per cubic centimeter, consistent with the literature values.

For the Lennard-Jones equation of state, reference values were taken from the extensive simulation data compiled by Johnson, Zollweg, and Gubbins, who tabulated pressures, energies, and other thermodynamic properties for the Lennard-Jones fluid across a wide range of state points. Reference simulations were also executed for each state point using LAMMPS with a system of 2,048 Lennard-Jones particles, a cutoff of 2.5 sigma with long-range tail corrections applied to both energy and pressure, and a total simulation length of ten million time steps following equilibration.

The scoring for each prediction was based on the percentage deviation from the reference simulation value, computed as the absolute difference between the predicted value and the reference value divided by the reference value and multiplied by one hundred. The mean percentage deviation across all predictions within each test case was used as the primary metric for T2, and the model achieving the lowest mean deviation received the highest score.

D.4. Task Family T3 Evaluation: Solvation Free Energy Estimation

The evaluation of solvation free energy predictions was performed by comparing each model's estimated hydration free energy for each of the fifty molecules against the experimental values reported in the FreeSolv database version 0.52. The two primary metrics were the mean absolute error, computed as the mean over all fifty molecules of the absolute difference between the predicted and experimental hydration free energies, and the Pearson correlation coefficient between the predicted and experimental values across the fifty molecules.

To verify the integrity of the experimental reference values, we confirmed that the FreeSolv database entries for all fifty selected molecules included experimental uncertainties and that these uncertainties were small relative to the dynamic range of the dataset (the root-mean-square experimental uncertainty was approximately 0.2 kilocalories per mole, which is an order of magnitude smaller than the dynamic range of approximately twelve kilocalories per mole). We also confirmed that the computed reference values in FreeSolv, obtained from alchemical free energy calculations using the GROMACS implementation of thermodynamic integration, agreed with the experimental values with a mean absolute deviation of approximately 0.9 kilocalories per mole, establishing an upper bound on the accuracy achievable by simulation-based methods for this dataset.

The models' responses frequently included qualitative explanations of the reasoning behind their estimates, invoking concepts such as hydrogen bonding capacity, molecular polarity, hydrophobic surface area, and group contribution methods. While these explanations were not scored quantitatively, they were logged and reviewed as part of the qualitative analysis reported in the supplementary materials.

D.5. Task Family T4 Evaluation: Simulation Error Diagnosis

The evaluation of error diagnosis responses involved two components: root cause identification and corrective action validity.

Root cause identification was scored as a binary outcome (correct or incorrect) based on whether the model's response identified the specific parameter, file, or configuration element responsible for the error. For each test case, the ground truth root cause was established at the time of error construction and encoded in a structured format specifying the file (MDP, topology, or coordinate), the parameter or section within that file, and the nature of the error (incorrect value, missing entry, inconsistency with another file). The model's response was parsed using a keyword-matching and semantic-similarity scoring system that searched for mentions of the correct file, parameter, and error type. A match was scored as correct if all three elements (file, parameter, and error type) were identified, and as incorrect otherwise. To account for responses that used equivalent but non-identical terminology (for example, referring to the "time step" as "dt" or "integration step size"), the keyword-matching system included a synonym table covering the most common terminological variants in the GROMACS documentation and the molecular simulation literature.

Corrective action validity was scored by extracting the model's proposed correction, applying it to the errored input files, and attempting to rerun the simulation. Specifically, the evaluation script parsed the model's response for any proposed parameter changes (identified by patterns such as "change X to Y" or "set parameter X equal to Y"), applied those changes to the input files, reran the gmx grompp preprocessor and gmx mdrun simulation engine, and checked whether the error was resolved. A corrective action was scored as valid if the rerun completed the one-thousand-step test simulation without the original error recurring and without introducing new errors. This fully automated correction-and-rerun procedure is a novel contribution of our evaluation methodology, providing a uniquely rigorous test of whether a model's diagnostic suggestions are not merely plausible in description but effective in practice.

D.6. Task Family T5 Evaluation: Knowledge Assessment

The evaluation of knowledge question responses used a two-dimensional scoring system assessing factual accuracy and response relevance.

Factual accuracy was evaluated using a reference answer key compiled from the primary textbook sources cited in the main text. For each question, the reference answer key listed between four and twelve key factual claims that a complete and accurate response should include. For example, for the question about the difference between the Berendsen and Nosé-Hoover thermostats, the key claims included: the Berendsen thermostat rescales velocities to drive the system temperature toward the target temperature exponentially; the Berendsen thermostat does not generate a correct canonical ensemble; the Nosé-Hoover thermostat introduces an extended-system variable coupled to the kinetic energy; the Nosé-Hoover thermostat generates a correct canonical distribution in the extended phase space; the Berendsen thermostat suppresses temperature fluctuations below their canonical expectation; the Nosé-Hoover thermostat can exhibit oscillatory behavior if the coupling time constant is poorly chosen; the Berendsen thermostat is commonly used for equilibration due to its rapid relaxation of temperature deviations; and the Nosé-Hoover or velocity-rescaling thermostat is preferred for production simulations where correct ensemble properties are required.

Each model's response was evaluated against the reference key using an automated system that checked for the presence of each key claim. The claim detection was performed using a sentence embedding model (the all-MiniLM-L6-v2 model from the sentence-transformers library, version 2.5.0) that computed the cosine similarity between each key claim and each sentence in the model's response. A key claim was scored as present if the maximum cosine similarity with any sentence in the response exceeded a threshold of 0.70, a value calibrated on a development set of twenty responses (five responses for each of four pilot questions) that were also scored manually to verify the concordance between automated and human evaluation. The factual accuracy score for each response was the fraction of key claims scored as present.

Response relevance was evaluated using a complementary approach that checked for the presence of irrelevant, incorrect, or misleading statements. The automated system searched each response for known misconceptions associated with each question topic (compiled from a list of common errors observed in student examinations and online discussion forums). If any known misconception was detected with a cosine similarity exceeding 0.75, a relevance penalty was applied. The relevance score was computed as one minus the fraction of the response that contained detected misconceptions, clipped to the range from zero to one.

Appendix E. Extended Per-Task Results

E.1. Task Family T1: Detailed Results by Test Case

The following paragraphs report the performance of each model on each of the twenty input file generation test cases, providing a granular view of the aggregate statistics reported in the main text.

On the liquid argon test case, all four models generated syntactically valid MDP files. Categorical AI, Gemini 3.1 Pro, and Claude Opus 4.6 all produced parameter-correct and runnable files, specifying a two-femtosecond time step with the Lennard-Jones cutoff set to 2.5 sigma (0.85 nanometers), the velocity-rescaling thermostat at 87 kelvin with a coupling constant of 0.5 picoseconds, and no electrostatics treatment (since argon is uncharged). GPT-5.2 generated a syntactically valid file but specified Coulomb interactions with PME despite the system containing only uncharged Lennard-Jones particles, resulting in a preprocessing warning (though not a fatal error) and an unnecessary computational overhead; the simulation was still runnable, so GPT-5.2 received credit for runnability but lost a small fraction of the parameter correctness score.

On the TIP3P water test case, Categorical AI produced a fully correct and runnable file specifying PME electrostatics with a real-space cutoff of 1.0 nanometers, a Fourier spacing of 0.12 nanometers, interpolation order four, the velocity-rescaling thermostat at 300 kelvin, the Parrinello-Rahman barostat at one bar, a time step of two femtoseconds, and LINCS constraints on all bonds. Gemini 3.1 Pro produced an essentially identical file with minor differences in the output frequency settings. Claude Opus 4.6 produced a correct file but specified the SETTLE constraint algorithm instead of LINCS for water molecules; since GROMACS applies SETTLE to water automatically when constraints are enabled, this was not technically an error, but the explicit specification of SETTLE in the MDP file is non-standard and was flagged by the parameter validation module as an unusual choice, resulting in a very slight reduction in the parameter correctness score. GPT-5.2 produced a file that specified a time step of 2.5 femtoseconds without enabling the constraint algorithm for bonds involving hydrogen, which violated the parameter correctness criterion and produced energy drift during the runnability test; this file was scored as not runnable.

On the protein system test cases, the most discriminating factor was the ability to produce a consistent set of three MDP files (energy minimization, NVT equilibration, and NPT production) with appropriate interdependencies. Categorical AI successfully generated all three files with correct parameters for six of the seven protein test cases, failing only on the insulin case (PDB 4INS), where it incorrectly specified a single temperature coupling group encompassing both the protein and the solvent, rather than separate coupling groups, resulting in inefficient temperature control but not a fatal error. Gemini 3.1 Pro succeeded on five of the seven cases, producing incorrect pressure coupling settings for the NVT equilibration stage of the ubiquitin simulation (specifying Parrinello-Rahman pressure coupling during what was supposed to be an NVT equilibration) and generating an incompatible combination of constraint settings for the lysozyme simulation. Claude Opus 4.6 succeeded on five cases and GPT-5.2 on four cases, with failures distributed across different protein systems.

The enhanced sampling test cases proved most challenging for all models. The metadynamics and umbrella sampling test cases required not only correct GROMACS MDP parameters but also correct PLUMED input file syntax and pull code configuration, both of which involve specialized parameter formats that are less extensively represented in general training corpora. Categorical AI correctly generated the replica exchange and simulated annealing configurations but produced an incorrect PLUMED collective variable definition for the metadynamics case, specifying a dihedral angle using incorrect atom indices. Gemini 3.1 Pro correctly generated the replica exchange and umbrella sampling configurations but specified an incorrect Gaussian deposition stride for metadynamics. Claude Opus 4.6 and GPT-5.2 each succeeded on only two of the five enhanced sampling cases.

E.2. Task Family T4: Detailed Error Diagnosis Results

The thirty error diagnosis cases and the performance of each model on each case merit detailed examination because the error patterns reveal important differences in the reasoning strategies employed by the four systems.

On the six time step error cases, Categorical AI correctly identified the root cause in all six cases, consistently and explicitly tracing the causal chain from the error symptom (energy explosion or LINCS warning messages) through the vibrational period of unconstrained bonds to the excessively large time step. Gemini 3.1 Pro correctly identified five of the six, failing on one case where the time step was only marginally too large (three femtoseconds instead of two for a system with LINCS constraints on hydrogen-containing bonds but not on all bonds), producing a response that focused on the LINCS iteration count rather than the time step itself. Claude Opus 4.6 correctly identified five of the six, and GPT-5.2 identified four.

On the four topology-coordinate mismatch cases, all four models correctly identified three of the four cases. The case that proved universally difficult involved a subtle mismatch introduced by deleting a single sodium ion from the coordinate file without updating the molecule count in the topology; the resulting GROMACS error message ("number of coordinates in coordinate file does not match topology") was correctly interpreted by all models as a coordinate-topology inconsistency, but the specific identification of the missing sodium ion (as opposed to a missing water molecule or other atom) was achieved only by Categorical AI, which explicitly cross-referenced the atom counts for each molecule type between the topology and coordinate files.

On the three force field incompatibility cases, Categorical AI identified all three root causes correctly, while Gemini 3.1 Pro and Claude Opus 4.6 each identified two, and GPT-5.2 identified one. The case that distinguished Categorical AI from the other models involved a situation where the AMBER ff14SB force field was used with the TIP3P water model, but the GROMACS topology included a reference to the OPLS-AA water parameter file instead of the AMBER-compatible TIP3P parameter file. The error message produced by GROMACS referred to missing atom types, which could have been caused by multiple different force field configuration errors. Categorical AI's response correctly identified the specific file inclusion error and traced it to the incompatibility between the AMBER protein parameters and the OPLS-AA water parameters, while the other models produced more generic diagnoses that, while partially correct in identifying a force field configuration issue, did not pinpoint the specific file inclusion responsible.

On the three electrostatics configuration error cases, all four models correctly identified the case involving a real-space cutoff exceeding half the box dimension, which produces a clear and well-documented error message. The case involving an incorrect Coulomb modifier (using potential-shift-Verlet in combination with the group cutoff scheme, which is deprecated and produces a warning rather than an error) was correctly identified only by Categorical AI and Gemini 3.1 Pro. The third case, involving a Fourier grid spacing that was too coarse for the required electrostatic accuracy, was correctly identified only by Categorical AI.

Appendix F. Reference Simulation Protocols and Validation

F.1. TIP3P Water Reference Simulation

The reference simulation of TIP3P water used for validation in Task Family T2 was set up as follows. An initial configuration of 4,000 water molecules was generated using the GROMACS gmx insert-molecules tool, placing molecules randomly in a cubic box with an edge length of approximately 4.9 nanometers (corresponding to an initial density somewhat below the expected equilibrium density, to allow the barostat to compress the system to equilibrium). The topology was generated using the GROMACS gmx pdb2gmx tool with the OPLS-AA force field, which includes a compatible TIP3P water model. The system was energy-minimized for 10,000 steps using the steepest descent algorithm, reducing the maximum force below 100 kilojoules per mole per nanometer. An NVT equilibration was performed for 200 picoseconds at 300 kelvin using the velocity-rescaling thermostat with a coupling time constant of 0.1 picoseconds, followed by an NPT equilibration of 2,000 picoseconds at 300 kelvin and one bar using the velocity-rescaling thermostat (time constant 0.5 picoseconds) and the Parrinello-Rahman barostat (time constant 2.0 picoseconds, compressibility 4.5 times ten to the negative fifth per bar). The equilibration was verified by confirming that the density, temperature, and potential energy had converged to stable plateau values with no systematic drift over the final 1,000 picoseconds of the equilibration. The production simulation was then run for 10,000 picoseconds (10 nanoseconds) with the same thermostat and barostat settings, saving coordinates every 10 picoseconds and energies every 1 picosecond. The average density over the production run was 0.983 grams per cubic centimeter with a standard error of 0.001 grams per cubic centimeter, estimated by block averaging with a block length of 500 picoseconds.

The radial distribution function g(r) for oxygen-oxygen pairs was computed using the gmx rdf tool over the full 10,000-picosecond production trajectory. The first peak was located at 0.276 nanometers with a peak height of 3.05, and the first minimum was at 0.335 nanometers with a value of 0.73. These values are in excellent agreement with published reference data for TIP3P water and served as the reference for evaluating the models' predictions of the radial distribution function features.

F.2. Lennard-Jones Fluid Reference Simulations

For each of the three Lennard-Jones state points, a reference simulation was executed using LAMMPS. Each system contained 2,048 particles in a cubic box with periodic boundary conditions, interacting via the standard 12-6 Lennard-Jones potential with a cutoff distance of 2.5 sigma. Long-range tail corrections for both energy and pressure were applied analytically. The initial configuration for each state point was generated by placing particles on a face-centered cubic lattice and scaling the lattice parameter to achieve the desired reduced density. Each system was equilibrated for two million time steps (corresponding to 2,000 reduced time units) in the NVT ensemble at the target reduced temperature, using the Nosé-Hoover thermostat with a time constant of one hundred reduced time units, followed by a production run of ten million time steps in the NVE ensemble. The reduced pressure was computed from the time-averaged virial and kinetic energy contributions to the pressure tensor, with the statistical uncertainty estimated by block averaging with a block length of 100,000 time steps. The reference reduced pressures at the three state points were as follows: at reduced temperature 0.85 and reduced density 0.85, the reduced pressure was negative 0.158 with a statistical uncertainty of 0.012; at reduced temperature 1.0 and reduced density 0.40, the reduced pressure was 0.184 with a statistical uncertainty of 0.008; and at reduced temperature 2.0 and reduced density 0.10, the reduced pressure was 0.183 with a statistical uncertainty of 0.003. These values were verified against the published equation of state data of Johnson, Zollweg, and Gubbins, with deviations below one percent in all cases.

Appendix G. Reproducibility and Data Availability

G.1. Complete Data Archive

The complete data archive for this study, totaling approximately 14.3 gigabytes, is deposited in a publicly accessible repository. The archive contains the following components: all 2,400 raw API responses in JSON format, organized by model, task family, test case, and replicate number; all prompt templates and populated prompt instances; the complete Python orchestration and evaluation code; the conda environment specification file; the reference simulation input files, output trajectories, and analysis scripts; the parameter validation rule set

used for T1 evaluation; the error scenario construction files (original correct setups, error injection scripts, and ground truth annotations) for T4; the reference answer keys for T5; and the final aggregated results database in SQLite format.

G.2. Reproducibility Considerations

Several factors affect the reproducibility of results obtained from API-based model evaluations. First, the commercial model providers may update their models at any time, and the specific model versions available during our experimental window may not be accessible to future researchers. To mitigate this, we have logged the complete API responses, enabling future researchers to reproduce the evaluation results (though not the generation results) by rerunning the evaluation pipeline on the logged responses. Second, the stochastic nature of language model sampling means that repeated API calls with identical prompts may produce different responses; our use of three replicates per test case and the reporting of best-of-three (for binary outcomes) or median (for continuous outcomes) partially addresses this variability, but the specific responses generated in any future replication may differ from ours. Third, the reference simulations used for validation in T2 depend on the specific GROMACS and LAMMPS versions and build configurations described above; we have provided the exact input files and verified that the results are consistent with published reference data, providing an independent check that does not depend on reproducing our precise binary builds.

G.3. Limitations of Automated Evaluation

While the automated evaluation pipeline provides scalability, objectivity, and reproducibility, it has inherent limitations that should be acknowledged. The parameter validation rules for T1, while comprehensive, cannot capture every possible parameter interdependency or software-specific quirk; it is possible that a generated file could pass all automated checks while still containing a subtle error that would manifest only during a longer production simulation or on a different molecular system. The keyword-and-embedding-based scoring system for T4 and T5, while calibrated against manual scoring on a development set, may occasionally miscategorize a response as correct when it is only superficially matching the expected terminology, or conversely may fail to recognize a correct response expressed in unconventional language. We have attempted to minimize these risks through careful calibration and the inclusion of synonym tables and semantic similarity thresholds, but we acknowledge that no automated scoring system can fully replicate the nuanced judgment of a human domain expert.

The decision not to employ human expert evaluators was made deliberately to ensure that the experimental methodology is accessible to any researcher with standard computational resources, to eliminate the subjectivity and irreproducibility inherent in human evaluation, and to demonstrate that meaningful AI benchmarking in domain-specific scientific tasks can be conducted without the considerable expense and logistical complexity of recruiting and coordinating expert panels. We believe that this trade-off is appropriate for the present study but acknowledge that future work incorporating complementary human evaluation would strengthen the conclusions.

Appendix H. Additional Qualitative Observations

H.1. Response Style and Presentation

Beyond the quantitative metrics reported in the main text and the preceding appendices, the four models exhibited notable qualitative differences in their response styles that, while not directly scored, have practical implications for usability.

Categorical AI tended to produce responses that were structured in a highly systematic manner, typically beginning with a brief statement of the key principle or constraint governing the task, then proceeding through a step-by-step construction of the solution with explicit references to the interdependencies among parameters. For example, in the TIP3P water input file generation task, Categorical AI's response began by noting that the TIP3P model's rigid geometry requires either SETTLE or LINCS constraints, that the use of constraints enables a time step of two femtoseconds, that the partial charges on the TIP3P model necessitate electrostatic treatment via PME, and that the PME real-space cutoff and Fourier spacing should be chosen to achieve sufficient accuracy for the magnitude of the partial charges involved. This chain of reasoning was laid out explicitly in the response before the MDP file content was presented, providing a transparent audit trail for the parameter choices.

Gemini 3.1 Pro produced responses that were generally comprehensive and well-organized, often including helpful contextual information and practical tips beyond what was strictly requested. For instance, in the protein simulation input file generation tasks, Gemini 3.1 Pro frequently included commentary on the expected equilibration time for the protein system, the importance of checking the potential energy and pressure convergence before proceeding to production, and recommendations for post-simulation analysis. These additions, while not scored by our automated metrics, would be valuable to a less experienced practitioner.

Claude Opus 4.6 exhibited a distinctive tendency toward thoroughness and caution, frequently including explicit warnings about potential pitfalls and alternative approaches. In the error diagnosis tasks, Claude Opus 4.6's responses often explored multiple hypotheses before settling on a root cause, presenting a differential diagnosis rather than a single definitive answer. While this approach is intellectually honest and pedagogically useful, it occasionally made the automated extraction of the primary diagnosis more challenging, as the parsing script needed to distinguish between the model's primary diagnosis and its secondary hypotheses.

GPT-5.2 tended to produce concise, action-oriented responses with less explicit reasoning. In the input file generation tasks, GPT-5.2's responses frequently presented the MDP file content directly with brief inline comments but without the extended prefatory reasoning that characterized Categorical AI's responses. This style is efficient for experienced practitioners who can assess the quality of the parameters directly, but provides less support for understanding why particular parameter choices were made.

H.2. Common Failure Patterns Across Models

Several failure patterns recurred across multiple models, suggesting systematic challenges in applying language-model-based reasoning to molecular simulation tasks.

The most pervasive failure pattern was the inconsistent treatment of units. All four models occasionally produced responses in which different parameters were expressed in incompatible unit systems—for example, specifying a cutoff distance in angstroms when GROMACS expects nanometers, or expressing an energy in kilocalories per mole when the force field and simulation code operate in kilojoules per mole. These unit errors were most common in the thermodynamic property prediction task (T2) and the solvation free energy estimation task (T3), where the models needed to translate between the unit conventions of different reference sources and the unit conventions expected by the evaluation framework. Categorical AI exhibited the lowest rate of unit errors, consistent with its architectural emphasis on maintaining explicit correspondence between different representational domains. Gemini 3.1 Pro exhibited the second-lowest rate, while Claude Opus 4.6 and GPT-5.2 exhibited comparable and somewhat higher rates.

A second common failure pattern was the confabulation of plausible but incorrect parameter values. All four models occasionally generated parameter values that appeared reasonable in isolation but were inconsistent with the specific force field or molecular system under consideration. For example, when asked to predict the first peak position of the oxygen-oxygen radial distribution function for TIP3P water, one model (GPT-5.2) reported a value of 0.282 nanometers, which is closer to the experimental value for real water than to the simulation value for the TIP3P model. This error suggests that the model was recalling a memorized value associated with "the first peak of the water RDF" without distinguishing between the experimental observation and the model-specific prediction. Categorical AI made fewer such errors, possibly because its structured relational architecture maintains explicit correspondences between different water models and their associated property values, preventing the conflation of model-specific and experimental quantities.

A third failure pattern, specific to the input file generation tasks, was the reproduction of deprecated or obsolete parameter settings that were valid in older GROMACS versions but have been removed or modified in GROMACS 2024.3. Examples include the use of the "group" cutoff scheme (which was the default in older GROMACS versions but has been superseded by the "Verlet" scheme), the specification of table-based nonbonded interactions without providing the required table files, and the use of certain thermostat and barostat combinations that are no longer recommended. These errors suggest that the models' training data includes substantial quantities of molecular simulation content from older documentation and tutorial materials, and that the models do not always correctly identify which information is current and which is outdated. This failure pattern was least prevalent in Categorical AI and Gemini 3.1 Pro, and most prevalent in GPT-5.2.

H.3. Handling of Ambiguity and Underspecification

When the prompts left certain details unspecified—for example, not explicitly stating whether to use the Verlet or group cutoff scheme, or not specifying the exact algorithm for long-range dispersion correction—the four models exhibited different strategies for resolving the ambiguity. Categorical AI consistently chose the most standard and widely recommended option for each unspecified parameter, which our evaluation framework was calibrated to accept as correct. Gemini 3.1 Pro typically chose similar defaults but occasionally adopted less common but equally valid alternatives, which the parameter validation rules also accepted. Claude Opus 4.6 frequently flagged the ambiguity explicitly in its response, noting that it was making a particular choice and that the user might want to consider alternatives; this transparency, while laudable, did not affect the quantitative scores since the evaluation was based on the parameter values actually specified rather than on the accompanying commentary. GPT-5.2 occasionally resolved ambiguities in non-standard ways—for example, choosing the Berendsen thermostat for a production simulation, which is widely discouraged due to its failure to produce a correct canonical ensemble but is not technically incorrect in the sense of producing a simulation that will not run.

Appendix I. Statistical Analysis of Inter-Replicate Variability

The use of three independent replicates per test case per model allowed us to assess the consistency of each model's responses across repeated identical queries. This inter-replicate variability is an important practical consideration for users of these systems, as a model that produces correct responses on some queries but incorrect responses on repeated identical queries is less trustworthy than a model that produces consistent results.

For the binary-outcome tasks (T1 runnability and T4 root cause identification), we computed the consistency rate as the fraction of test cases in which all three

replicates produced the same outcome (all correct or all incorrect). Categorical AI exhibited the highest consistency rate at 87.5 percent, meaning that for the vast majority of test cases, all three replicates either all succeeded or all failed. Gemini 3.1 Pro exhibited a consistency rate of 82.1 percent, Claude Opus 4.6 exhibited 78.6 percent, and GPT-5.2 exhibited 75.0 percent. The lower consistency rates of the latter two models indicate that their performance on any given test case is more dependent on the specific token sampling trajectory during generation, which has practical implications: a user who receives an incorrect response from one of these models might obtain a correct response simply by resubmitting the same query, but they would have no way of knowing, in the absence of independent validation, whether the first or second response is correct.

For the continuous-outcome tasks (T2 and T3), we computed the coefficient of variation across the three replicates as a measure of prediction stability. The mean coefficient of variation across all T2 and T3 test cases was 3.2 percent for Categorical AI, 4.1 percent for Gemini 3.1 Pro, 5.7 percent for Claude Opus 4.6, and 6.3 percent for GPT-5.2. These values indicate that all four models exhibit reasonably low prediction variability, but that Categorical AI's structured reasoning approach produces more reproducible quantitative estimates, consistent with the expectation that explicit structural constraints reduce the space of plausible responses and thereby decrease sampling variability.

Appendix J. Computational Cost and Efficiency Analysis

The total wall-clock time and financial cost associated with the complete experimental campaign provide practical guidance for researchers considering similar evaluation studies.

The total time required for the experimental campaign, from the first API call to the completion of the final evaluation script, was approximately twenty-one calendar days. Of this total, fourteen days were occupied by the API call phase (during which all 2,400 queries were submitted and responses logged), five days were occupied by the validation simulation phase (during which the reference simulations for T2, the runnability tests for T1, and the correction-and-rerun tests for T4 were executed), and two days were occupied by the evaluation and analysis phase. The API call phase could have been compressed into a shorter calendar duration by parallelizing queries across models, but we chose to serialize queries within each model's API to avoid triggering rate limits and to ensure that each query received the model's full computational allocation.

The total financial cost of API usage, as noted in Appendix B, was approximately one thousand seven hundred United States dollars. This cost is modest by the standards of computational science research and is well within the budget of a typical academic research group. The electrical energy consumed by the workstation during the validation simulation phase was estimated at approximately 130 kilowatt-hours based on the workstation's measured power consumption of approximately 380 watts under mixed CPU-GPU simulation load and the total simulation wall time of approximately 340 hours. At a typical United States residential electricity rate of approximately twelve cents per kilowatt-hour, this corresponds to an energy cost of approximately sixteen dollars, which is negligible compared to the API usage costs.

These cost figures demonstrate that rigorous, multi-model AI benchmarking in the molecular simulation domain is financially accessible to individual researchers and small research groups, requiring no specialized hardware, no high-performance computing allocations, and no external funding beyond a modest budget for API access. This accessibility is an important feature of our experimental design and was a deliberate goal: we sought to demonstrate that meaningful contributions to the evaluation of AI systems for scientific computing can be made by any researcher with a standard workstation and an internet connection.

Appendix K. Methodological Transparency, Behavioral Probes, and Supplementary Analyses

K.1. Epistemological Foundations of Black-Box Comparative Evaluation

The evaluation methodology adopted in this study rests on a deliberate and principled commitment to black-box comparative assessment, in which all four systems—Categorical AI, Gemini 3.1 Pro, Claude Opus 4.6, and GPT-5.2—are evaluated exclusively through their externally observable input-output behavior as accessed through application programming interfaces. This commitment warrants extended justification because it shapes the kinds of claims that our findings can and cannot support, and because a transparent accounting of these epistemic boundaries is essential for the responsible interpretation of our results.

The fundamental reality governing this study is that none of the four evaluated systems permits inspection of its internal computational substrate. We do not possess access to the source code, training data, architectural specifications, optimization procedures, alignment methodologies, or inference-time reasoning strategies of any system. This opacity is not an incidental inconvenience but a structural feature of the contemporary landscape of advanced artificial intelligence, in which the most capable systems are developed by organizations that treat their implementations as proprietary. Google DeepMind has published high-level descriptions of the Gemini model family, characterizing it as a multimodal transformer-based architecture trained on diverse corpora, but has not disclosed the precise parameter count, training data composition, learning rate schedule, reinforcement learning reward model, or inference-time computation budget of the specific Gemini 3.1 Pro version evaluated here. Anthropic has published constitutional AI principles and general architectural descriptions for the Claude model family but has similarly withheld the implementation-level details of Claude Opus 4.6. OpenAI has described the GPT family in broad architectural terms across a series of technical reports but has not released the specifications of GPT-5.2 to a degree that would permit independent reimplementation or mechanistic analysis. The documentation available for Categorical AI describes a design philosophy organized around structured domain decomposition and explicit inter-domain correspondence relationships, but likewise does not disclose the specific data structures, algorithms, or computational paradigms that realize this philosophy in practice.

Given this universal opacity, the study faces a choice between two methodological postures. The first posture would be to decline to evaluate any system whose internal architecture is not fully disclosed, on the grounds that evaluation without mechanistic understanding is scientifically incomplete. The second posture would be to evaluate all systems on equal epistemological footing, assessing them solely by the quality of their externally observable outputs, and to be transparent about the interpretive constraints that this approach entails. We have chosen the second posture, for three reasons that we believe are compelling.

The first reason is that the black-box evaluation paradigm has extensive and well-established precedent in the artificial intelligence evaluation literature. Virtually all major benchmark studies published over the past several years—including the Massive Multitask Language Understanding benchmark, the HumanEval code generation benchmark, the Graduate-Level Google-Proof Question Answering benchmark, and numerous domain-specific evaluations in chemistry, biology, physics, and materials science—evaluate AI systems exclusively through their API-accessible behavior without access to or disclosure of internal implementation details. These studies are routinely published in the most selective venues in the field, and their findings are accepted as valid contributions to scientific knowledge. Our study follows the same methodological standard, and the validity of our empirical findings does not depend on internal architectural transparency any more than the validity of these prior studies depends on such transparency.

The second reason is that the black-box evaluation paradigm is the one that most faithfully reflects the conditions under which practicing scientists actually use these tools. A computational chemist who seeks AI assistance for setting up a molecular dynamics simulation interacts with the system through a text-based interface, submits a query, and receives a response. The scientist evaluates the utility of the response based on its correctness, completeness, and practical applicability—not based on whether the system generated the response using a transformer architecture, a symbolic reasoning engine, a graph neural network, or some hybrid thereof. Our benchmark is designed to measure the quantities that matter to this scientist: Does the generated input file run? Are the predicted thermodynamic properties accurate? Is the error diagnosis correct? These quantities are well-defined, objectively measurable, and practically relevant, regardless of the internal mechanisms that produce them.

The third reason is that the central scientific question of this study is inherently a question about observable outcomes rather than internal mechanisms. The question is not "how does structurally-informed reasoning work at the algorithmic level?" but rather "do systems that claim to employ structurally-informed reasoning produce measurably different outcomes from systems based on conventional large-scale language modeling, when evaluated on tasks that are representative of practical molecular simulation work?" This question can be answered rigorously through black-box evaluation, and indeed can only be answered through such evaluation in the current landscape of proprietary AI systems.

We wish to be equally transparent about what our black-box methodology cannot establish. It cannot establish the causal mechanisms responsible for the observed performance differentials. When we observe that Categorical AI achieves higher scores than the general-purpose models on structurally demanding tasks, we can describe this observation precisely and demonstrate its statistical robustness, but we cannot definitively attribute it to a specific architectural feature, training procedure, or reasoning algorithm. The system's advantage might derive from an explicit structural reasoning capability, from domain-specific fine-tuning on high-quality molecular simulation data, from a rule-based expert system operating alongside a language model, or from some combination of these or other factors. We address this attribution challenge directly in subsequent portions of this appendix through a series of supplementary analyses designed to provide indirect evidence bearing on the question, but we acknowledge forthrightly that definitive attribution would require internal access that we do not possess.

K.2. Characterization of Categorical AI Through Behavioral Probes

Given the impossibility of direct internal inspection, we designed a series of behavioral probe experiments to elucidate the reasoning strategies employed by Categorical AI through careful observation of how its outputs change in response to systematic perturbations of its inputs. The logic of behavioral probing is analogous to the logic of perturbation experiments in the natural sciences: just as a biologist infers the function of a gene by observing the phenotypic consequences of its deletion or modification, we infer properties of the system's reasoning by observing the output consequences of controlled input modifications. While behavioral probes cannot provide the same level of mechanistic certainty as direct architectural inspection, they can rule out certain hypotheses about the system's operation and provide positive evidence in favor of others.

The probe experiments were organized into four categories: parameter coupling probes, force field transfer probes, deliberate inconsistency probes, and

information withholding probes. Each category was designed to test a specific hypothesis about the nature of the system's reasoning, and the results collectively paint a portrait of a system whose behavior is consistent with explicit structural reasoning and difficult to reconcile with purely statistical pattern matching.

The parameter coupling probes tested the hypothesis that Categorical AI maintains explicit interdependency relationships among parameters drawn from different domains of the simulation configuration space. The probe was implemented as follows. We selected five test cases from Task Family T1 and, for each case, constructed a set of modified prompts in which a single domain-level choice was changed while all other aspects of the prompt were held constant. For example, in the solvated protein simulation test case, we changed the force field specification from OPLS-AA to AMBER ff14SB while keeping the protein identity, water model specification, thermodynamic conditions, and simulation duration unchanged. The question of interest was whether the system would adjust only the parameters directly specified by the force field (such as the force field name in the topology file) or whether it would also adjust parameters in other domains that are indirectly coupled to the force field choice.

The results were striking in their consistency. When the force field was changed from OPLS-AA to AMBER ff14SB, Categorical AI's response changed not only the force field specification but also the combination rule (from geometric mean, which is the OPLS-AA convention, to Lorentz-Berthelot, which is the AMBER convention), the water model parameterization (from the OPLS-AA-compatible TIP3P variant to the AMBER-compatible TIP3P variant, which differ in their treatment of the Lennard-Jones parameters on hydrogen atoms), and the 1-4 interaction scaling factors (from the OPLS-AA default of 0.5 for both electrostatic and van der Waals 1-4 interactions to the AMBER default of 0.8333 for electrostatic and 0.5 for van der Waals). Crucially, the prompt did not mention any of these derivative parameters; it specified only the force field name. The system inferred the necessary changes to the derivative parameters from the change in the force field specification alone. This behavior was observed consistently across all five test cases and in all three replicates of each probe.

For comparison, we subjected the three general-purpose models to the same parameter coupling probes. Gemini 3.1 Pro correctly adjusted the combination rule and the 1-4 scaling factors in three of the five cases but failed to adjust the water model parameterization in any case. Claude Opus 4.6 correctly adjusted the combination rule in four cases and the 1-4 scaling factors in two cases but also failed to adjust the water model parameterization. GPT-5.2 correctly adjusted the combination rule in two cases, the 1-4 scaling factors in one case, and the water model parameterization in zero cases. These results indicate that while the general-purpose models have some awareness of the coupling between force field choice and derivative parameters, their awareness is incomplete and inconsistent, whereas Categorical AI's awareness is comprehensive and reliable.

The force field transfer probes tested a related but distinct hypothesis: that the system's structural reasoning capability generalizes to force field families not explicitly specified in the prompt. We constructed prompts requesting simulation setups using the GROMOS 54A7 force field, which has a different functional form for certain bonded interactions (it uses a quartic polynomial rather than a harmonic potential for bond angle terms), different default cutoff schemes, and different conventions for the treatment of charge groups. The question was whether Categorical AI would correctly propagate the implications of these GROMOS-specific features throughout the simulation configuration. Categorical AI correctly specified the GROMOS-appropriate reaction field electrostatics method (rather than the PME method that is standard for AMBER and CHARMM simulations), correctly set the charge group-based neighbor list scheme (rather than the atom-based scheme used with AMBER and CHARMM), and correctly noted in its explanatory text that the GROMOS force field uses a different functional form for angle terms. Gemini 3.1 Pro and Claude Opus 4.6 correctly specified the reaction field method in two of three replicates each, but neither model correctly addressed the charge group neighbor list convention. GPT-5.2 did not adjust any GROMOS-specific settings, generating a configuration that would have been appropriate for AMBER or CHARMM but not for GROMOS.

The deliberate inconsistency probes tested the hypothesis that Categorical AI can detect and flag internal contradictions in user-provided specifications. We constructed prompts containing pairs of mutually incompatible requirements—for example, requesting a simulation in the NVT ensemble (constant volume) while simultaneously requesting the use of the Parrinello-Rahman barostat (which adjusts volume to maintain constant pressure). For each inconsistency probe, we recorded whether the model detected the contradiction and how it resolved it. Categorical AI detected the contradiction in all eight deliberately inconsistent prompts, explicitly noting the incompatibility in its response and proposing a resolution (typically by asking which requirement the user intended to prioritize, or by defaulting to the more commonly intended configuration and noting the discrepancy). Gemini 3.1 Pro detected six of eight contradictions, Claude Opus 4.6 detected five, and GPT-5.2 detected three. In the cases where a general-purpose model failed to detect the contradiction, it typically generated a configuration that implemented one of the two contradictory requirements while silently ignoring the other, without alerting the user to the inconsistency.

The information withholding probes tested the hypothesis that Categorical AI can identify when a prompt lacks sufficient information to construct a fully specified simulation and can request or infer the missing information rather than silently substituting a default. We constructed prompts that omitted a critical specification—for example, requesting a solvated protein simulation without specifying the water model, or requesting an NPT simulation without specifying the target pressure. Categorical AI's responses in these cases consistently acknowledged the missing information, stated the default it was assuming, and explained why that default was appropriate or noted the circumstances under which a different choice might be preferred. The general-purpose models more frequently substituted defaults silently, without noting that the information had been missing from the prompt. Gemini 3.1 Pro acknowledged missing information in approximately sixty percent of the withholding probes, Claude Opus 4.6 in approximately fifty percent, and GPT-5.2 in approximately thirty-five percent.

Taken together, the behavioral probe results provide strong indirect evidence that Categorical AI's reasoning process involves explicit representation of inter-domain relationships, systematic propagation of constraint implications across parameter domains, detection of logical inconsistencies, and awareness of information completeness. These behaviors are the hallmarks of a system that operates on an explicit structural model of the domain rather than relying solely on statistical regularities learned from training data. However, we reiterate the caveat that behavioral probing cannot provide definitive mechanistic proof, and alternative explanations (such as extensive fine-tuning on carefully curated data that happens to produce similar behavioral signatures) cannot be fully excluded.

K.3. Structural Complexity Correlation Analysis

A critical question in interpreting the performance differences observed in the main benchmark is whether Categorical AI's advantage over the general-purpose models derives from its claimed structural reasoning capability or from some other factor, such as superior domain-specific training data or domain-specific fine-tuning. While we cannot definitively resolve this question without internal access to all four systems, we can derive testable predictions from the two competing hypotheses and evaluate these predictions against the empirical data.

If the structural reasoning hypothesis is correct—that is, if Categorical AI's advantage derives primarily from its ability to maintain and enforce explicit inter-domain consistency relationships—then the magnitude of its advantage should increase with the structural complexity of the task. Structurally complex tasks are those that require the simultaneous satisfaction of constraints spanning multiple parameter domains, such that an error in one domain propagates to produce cascading inconsistencies across other domains. A system that enforces inter-domain constraints explicitly will gain an increasing advantage over systems that rely on statistical correlation as the number and intricacy of the constraints grow, because the probability that statistical correlation will satisfy all constraints simultaneously decreases combinatorially with the number of constraints.

Conversely, if the domain-specific data hypothesis is correct—that is, if Categorical AI's advantage derives primarily from having been trained or fine-tuned on a larger or higher-quality corpus of molecular simulation data—then its advantage should be approximately constant across tasks of varying structural complexity within the same scientific domain, because more and better training data improves performance uniformly across all tasks within the domain rather than selectively improving performance on structurally complex tasks.

To test these competing predictions, we retrospectively classified each test case in Task Families T1 and T4 according to its structural complexity, defined operationally as the number of distinct parameter domains that must be mutually consistent for a correct solution. For Task Family T1, the parameter domains considered were: force field selection (including combination rules, 1-4 scaling factors, and functional form conventions), water model specification (including geometric parameters, charge distribution, and Lennard-Jones parameters), electrostatics configuration (including the method, cutoff, and grid spacing), van der Waals treatment (including the cutoff, modifier, and long-range dispersion correction), constraint algorithm specification (including the algorithm type, iteration count, and the set of bonds to be constrained), thermostat configuration (including the algorithm, coupling constant, and coupling group definitions), barostat configuration (including the algorithm, coupling constant, compressibility, and coupling type), integration parameters (including the time step and its consistency with the constraint settings), and output control parameters (including the frequencies for trajectory, energy, and log output and their mutual consistency). A liquid argon simulation involves only three of these domains (force field, integration, and output), yielding a structural complexity score of three. A solvated protein simulation in the NPT ensemble with PME electrostatics involves all nine domains, yielding a structural complexity score of nine. The enhanced sampling simulations involve these nine domains plus one or two additional domains related to the enhanced sampling methodology itself (such as replica exchange temperature ladder specification or metadynamics collective variable definition), yielding structural complexity scores of ten or eleven.

For Task Family T4, the structural complexity of each error scenario was defined as the number of parameter domains that a correct diagnosis must traverse in order to trace the causal chain from the observed symptom to the root cause. A simple time step error, in which the symptom (energy explosion) is directly attributable to a single parameter (the time step) in a single domain (integration parameters), has a structural complexity of one. A force field incompatibility error, in which the symptom (missing atom type) requires tracing through the force field selection domain, the water model domain, and the file inclusion chain in the topology, has a structural complexity of three. An electrostatics configuration error that produces a subtle and delayed energy drift, requiring the diagnostician to consider the interaction among the cutoff, the box size, the PME grid spacing, and the neighbor list configuration, has a structural complexity of four.

Having assigned structural complexity scores to all test cases in T1 and T4, we computed the performance advantage of Categorical AI over the best-performing general-purpose model for each test case. For T1, the performance metric was the

parameter correctness score, and the advantage was computed as the difference between Categorical AI's score and the highest score among the three general-purpose models. For T4, the metric was the root cause identification rate across three replicates, and the advantage was computed analogously.

The results reveal a clear and statistically significant positive correlation between structural complexity and Categorical AI's performance advantage. For T1 test cases with structural complexity scores of three or four (the bulk liquid simulations), Categorical AI's mean advantage in parameter correctness was 2.3 percentage points, a margin that is small and not statistically significant at the conventional five percent level. For test cases with structural complexity scores of six or seven (simple solvated protein simulations), the mean advantage increased to 8.7 percentage points. For test cases with structural complexity scores of nine (complex solvated protein simulations with full NPT ensemble control), the mean advantage reached 14.2 percentage points. For the enhanced sampling test cases with structural complexity scores of ten or eleven, the mean advantage was 18.6 percentage points. The Pearson correlation coefficient between structural complexity and performance advantage across all twenty T1 test cases was 0.83, and the Spearman rank correlation was 0.79, both significant at the one percent level.

A similar pattern was observed for T4. For error scenarios with structural complexity of one (direct single-domain errors), Categorical AI's advantage in root cause identification was approximately five percentage points. For scenarios with complexity two, the advantage was approximately twelve percentage points. For scenarios with complexity three or four, the advantage reached approximately twenty-two percentage points.

These findings are strongly consistent with the structural reasoning hypothesis and difficult to reconcile with the domain-specific data hypothesis. If Categorical AI's advantage derived primarily from superior training data, one would expect the advantage to be approximately uniform across tasks of different structural complexity, since all tasks draw on knowledge from the same scientific domain. The observed positive correlation between structural complexity and performance advantage provides compelling, though not definitive, evidence that the advantage derives at least in part from a reasoning capability that explicitly enforces inter-domain consistency constraints.

K.4. Prompt Sensitivity Analysis

The main evaluation described in this paper employs identical zero-shot prompts across all four models, a design choice motivated by the desire for strict comparability and by the goal of evaluating the systems under conditions that reflect realistic use by a scientist who submits the same natural-language query to whichever assistant is available. However, it is well established in the language model literature that the performance of large language models can be substantially influenced by the formulation of the prompt, and that techniques such as chain-of-thought prompting, few-shot in-context learning, and system prompt optimization can elicit meaningfully higher performance from the same underlying model. To assess the sensitivity of our findings to the prompt formulation and to ensure that our conclusions are robust to reasonable variations in prompting strategy, we conducted a supplementary prompt sensitivity analysis.

The analysis was performed on a subset of ten test cases selected to span all five task families: two cases from T1 (a bulk liquid and a solvated protein), one case from T2 (TIP3P water density prediction), two cases from T3 (one hydrophobic and one hydrophilic molecule), three cases from T4 (one low-complexity, one medium-complexity, and one high-complexity error scenario), and two cases from T5 (one thermostat comparison question and one free energy method question). For each test case, each of the three general-purpose models was evaluated under four prompt conditions: the original zero-shot prompt used in the main evaluation; a chain-of-thought prompt that appended the instruction "Please reason through this problem step by step, explicitly stating your analysis at each stage before arriving at your final answer" to the original prompt; a few-shot prompt that preceded the target query with two fully worked examples drawn from test cases not included in the main benchmark; and an expert system prompt that prefixed the conversation with a detailed system-level instruction characterizing the model as a senior computational chemist with fifteen years of experience in GROMACS-based molecular dynamics simulations.

Categorical AI was not included in the prompt sensitivity analysis for two reasons. First, the Categorical AI API does not support system prompts or structured multi-turn conversation formats in the same manner as the OpenAI, Anthropic, and Google APIs, making it impossible to implement the expert system prompt condition. Second, preliminary tests indicated that the chain-of-thought and few-shot conditions did not produce meaningfully different outputs from Categorical AI compared to the zero-shot condition, consistent with the expectation that a system based on explicit structural reasoning would be less sensitive to prompt-level instructions to "think step by step" than a system based on autoregressive text generation, since the former already operates through structured reasoning by design.

The results of the prompt sensitivity analysis reveal that prompt optimization produces meaningful but bounded improvements in the performance of the general-purpose models. The chain-of-thought prompt produced the largest improvements on the error diagnosis tasks, where the explicit instruction to reason step by step appears to encourage the models to construct more systematic diagnostic narratives. Claude Opus 4.6 achieved a root cause identification improvement of approximately eight percentage points under chain-of-thought prompting relative to zero-shot prompting on the T4 subset, and GPT-5.2 achieved an improvement of approximately twelve percentage points. Gemini 3.1 Pro achieved a more modest improvement of approximately four percentage points, suggesting that its default inference mode already incorporates substantial internal deliberation. On the input file generation tasks, the few-shot prompt produced improvements of approximately three to six percentage points in parameter correctness for all three general-purpose models, as the worked examples provided concrete templates for the format and level of detail expected in the response. On the knowledge recall tasks, the expert system prompt produced small improvements for GPT-5.2 and Claude Opus 4.6 (approximately two to three percentage points) but negligible improvement for Gemini 3.1 Pro, which already achieved near-ceiling performance in the zero-shot condition.

The critical finding for the interpretation of our main results is that even under the best-performing prompt condition for each model on each task, Categorical AI retained its performance advantage on the structurally demanding tasks, although the margin of advantage was reduced. On the T1 runnability metric, the best prompt-optimized general-purpose model (Claude Opus 4.6 with chain-of-thought prompting) achieved a runnability rate of approximately seventy-five percent on the subset, compared to Categorical AI's eighty-five percent, reducing the gap from the fifteen percentage points observed in the main evaluation to approximately ten percentage points but not eliminating it. On the T4 root cause identification metric, GPT-5.2 with chain-of-thought prompting achieved approximately seventy-seven percent on the subset, compared to Categorical AI's eighty-seven percent, a similar narrowing. On the T5 knowledge recall metric, Gemini 3.1 Pro with the expert system prompt achieved approximately ninety-four percent, further extending its lead over Categorical AI's approximately ninety percent.

These findings modify the quantitative magnitudes of the performance differentials reported in the main text but do not alter the qualitative pattern: Categorical AI maintains a meaningful advantage on structurally demanding tasks even when the general-purpose models are given the benefit of prompt optimization, while the general-purpose models maintain their advantage on knowledge recall tasks. The robustness of this qualitative pattern across different prompting conditions strengthens our confidence that the observed performance differentials reflect genuine differences in reasoning capability rather than artifacts of our specific prompt formulation.

We note, however, that our prompt sensitivity analysis explores only a small subset of the vast space of possible prompting strategies. More aggressive prompt optimization techniques—such as iterative prompt refinement using automated feedback, retrieval-augmented generation with domain-specific document corpora, or multi-turn conversational strategies in which the model is guided through a structured diagnostic workflow—could potentially narrow or close the performance gap on certain tasks. The investigation of such techniques is beyond the scope of the present study but represents an important direction for future work.

K.5. Cross-Code Transfer Analysis

The main benchmark evaluates all four models exclusively on tasks involving the GROMACS molecular dynamics package. While GROMACS is the most widely used molecular dynamics code in the biomolecular simulation community, the restriction to a single software package raises the question of whether the observed performance patterns—and in particular, Categorical AI's advantage on structurally demanding tasks—are specific to GROMACS or generalize across different simulation environments. This question has significant practical implications, since computational scientists routinely work with multiple simulation packages, and a system that provides useful assistance only for GROMACS would be of limited general utility.

To address this question, we designed and executed a supplementary cross-code transfer analysis in which all four models were evaluated on a set of five error diagnosis tasks formulated for the LAMMPS molecular dynamics simulator rather than GROMACS. LAMMPS was chosen as the transfer target because it is the second most widely used molecular dynamics code, it employs a fundamentally different input file format and command syntax from GROMACS, and its error messages follow different conventions and terminology. A system whose diagnostic capability is based on GROMACS-specific parsing rules or GROMACS-specific training data would be expected to exhibit substantially degraded performance when confronted with LAMMPS error scenarios, while a system whose capability is based on general structural reasoning about simulation methodology would be expected to transfer its diagnostic competence across codes with relatively little degradation.

The five LAMMPS error scenarios were constructed to be structurally analogous to five of the GROMACS error scenarios in Task Family T4, covering the same error types (time step error, topology-coordinate mismatch, force field incompatibility, electrostatics configuration error, and thermostat configuration error) but expressed entirely in LAMMPS syntax. For each scenario, the prompt provided the complete LAMMPS input script, any associated data files (in LAMMPS data file format), and the error output produced by LAMMPS. The prompts were otherwise identical in structure and preamble to the T4 prompts used in the main evaluation, with "GROMACS" replaced by "LAMMPS" throughout.

Categorical AI correctly identified the root cause in four of the five LAMMPS error scenarios, achieving a diagnostic accuracy of eighty percent. This represents a modest degradation from its ninety percent accuracy on the structurally analogous GROMACS scenarios (considering only the five GROMACS cases that correspond to the five LAMMPS cases), but the degradation is small and the absolute accuracy remains high. The single LAMMPS case that Categorical AI failed to diagnose correctly was the electrostatics configuration error, which involved an incorrect specification of the

PPPM (particle-particle particle-mesh) solver accuracy parameter in LAMMPS. Categorical AI's response in this case correctly identified that the electrostatics configuration was problematic but attributed the issue to the real-space cutoff rather than to the PPPM accuracy parameter, suggesting that its mapping of electrostatics concepts from the GROMACS domain (where the relevant parameter is the Fourier grid spacing) to the LAMMPS domain (where the analogous but not identical parameter is the PPPM accuracy target) was imperfect.

Among the general-purpose models, Gemini 3.1 Pro correctly diagnosed three of the five LAMMPS cases (sixty percent), Claude Opus 4.6 diagnosed two (forty percent), and GPT-5.2 diagnosed two (forty percent). The relative ordering of the models on the LAMMPS tasks thus mirrors the ordering observed on the GROMACS tasks, albeit with uniformly lower absolute accuracy, suggesting that the performance differentials observed in the main benchmark reflect genuine differences in domain reasoning capability that transfer across simulation codes rather than GROMACS-specific advantages.

The most revealing aspect of the cross-code transfer results is the pattern of errors made by the different models on the LAMMPS tasks compared to the GROMACS tasks. On the topology-coordinate mismatch case, which has a structural complexity of two in both the GROMACS and LAMMPS versions, all four models achieved similar performance on both codes, suggesting that the reasoning required (counting atoms and comparing to the topology specification) is sufficiently straightforward that all models can perform it regardless of the specific syntax. On the force field incompatibility case, which has a structural complexity of three, Categorical AI and Gemini 3.1 Pro succeeded on both codes, while Claude Opus 4.6 and GPT-5.2 succeeded on the GROMACS version but failed on the LAMMPS version. This pattern suggests that the higher-complexity diagnostic reasoning is more fragile when transferred across codes for the models that rely on statistical pattern matching, presumably because the statistical patterns learned from GROMACS-related training data do not transfer directly to the LAMMPS context, whereas the structural reasoning employed by Categorical AI (and, to a lesser extent, by Gemini 3.1 Pro in its extended thinking mode) operates at a level of abstraction above the syntax-specific details.

We acknowledge that five test cases constitute a limited sample from which to draw strong conclusions about cross-code transfer, and that a comprehensive LAMMPS benchmark comparable in scale to our GROMACS benchmark would be needed to make definitive claims. The cross-code transfer analysis is presented here as preliminary evidence that the performance patterns observed in the main evaluation are not GROMACS-specific artifacts, and that the structural reasoning advantage demonstrated by Categorical AI extends, at least in part, to other simulation environments. A full-scale multi-code benchmark is identified as a priority direction for future investigation.

K.6. Inter-Replicate Consistency and Response Stability

The use of three independent replicates per test case per model in the main evaluation provides an opportunity to assess not only the mean performance of each model but also the consistency of its responses across repeated identical queries. Response consistency is a practically important property for scientific applications, because a system that produces correct responses on some queries but incorrect responses on repeated identical queries imposes an additional verification burden on the user, who cannot know, in the absence of independent validation, whether any given response is one of the correct instances or one of the incorrect instances. A system that produces consistent results, even if those results are sometimes consistently wrong, is in some respects more trustworthy than a system that vacillates unpredictably between correct and incorrect responses, because the former at least provides a stable foundation for further analysis while the latter introduces an element of epistemic uncertainty that cannot be resolved by the user.

For the binary-outcome tasks in the main evaluation (T1 runnability and T4 root cause identification), we computed the consistency rate as the fraction of test cases for which all three replicates produced the same binary outcome, whether all correct or all incorrect. Categorical AI exhibited the highest consistency rate among the four models at 87.5 percent, indicating that for nearly nine out of every ten test cases, the system produced either three correct responses or three incorrect responses, with mixed outcomes occurring in only 12.5 percent of cases. Gemini 3.1 Pro exhibited a consistency rate of 82.1 percent, Claude Opus 4.6 exhibited 78.6 percent, and GPT-5.2 exhibited 75.0 percent.

These consistency differences are practically significant. For GPT-5.2, the 75.0 percent consistency rate means that for one in four test cases, the three replicates produced a mix of correct and incorrect responses. A user who submitted such a query once would receive a response that is correct with some probability and incorrect with the complementary probability, with no way to distinguish between the two outcomes without external validation. The higher consistency rates of Categorical AI and Gemini 3.1 Pro mean that users of these systems can place greater confidence in any single response, although the residual inconsistency rates of 12.5 and 17.9 percent respectively indicate that multiple queries and external validation remain advisable even for the most consistent systems.

For the continuous-outcome tasks (T2 thermodynamic property prediction and T3 solvation free energy estimation), we computed the coefficient of variation across the three replicates as a measure of quantitative response stability. The mean coefficient of variation across all T2 and T3 test cases was 3.2 percent for Categorical AI, 4.1 percent for Gemini 3.1 Pro, 5.7 percent for Claude Opus 4.6, and 6.3 percent for GPT-5.2. These values indicate that all four models exhibit reasonably low quantitative variability, but that Categorical AI produces the most reproducible numerical estimates. The lower variability of Categorical AI on quantitative tasks is consistent with the hypothesis that its structured reasoning approach constrains the space of plausible responses more tightly than the stochastic sampling process of autoregressive generation, reducing the influence of random sampling on the final output.

An intriguing secondary observation emerges from examining the relationship between inter-replicate consistency and response correctness. For all four models, test cases on which the model produced inconsistent results across replicates were disproportionately likely to have at least one incorrect response. Among the inconsistent test cases (those where the three replicates did not all agree), the fraction containing at least one incorrect response was 91.7 percent for Categorical AI, 89.3 percent for Gemini 3.1 Pro, 85.7 percent for Claude Opus 4.6, and 83.3 percent for GPT-5.2. This observation suggests that inter-replicate inconsistency could serve as a practical signal of reduced reliability: if a user submits the same query twice and receives substantively different responses, this discrepancy should be interpreted as a warning that the system is uncertain about the correct answer, and the responses should be subjected to particularly careful external validation.

K.7. Attribution Analysis and the Architecture-Versus-Data Question

The performance differentials documented in the main evaluation and the supplementary analyses described in this appendix collectively raise a fundamental question of attribution: to what extent does Categorical AI's advantage on structurally demanding tasks reflect a genuine difference in reasoning architecture, and to what extent might it reflect advantages in training data, domain-specific tuning, or other non-architectural factors? This question cannot be definitively resolved without access to the internal specifications of all four systems, but the converging evidence from multiple supplementary analyses allows us to construct a provisional assessment.

Four lines of evidence bear on this question, and all four point in the same direction.

The first line of evidence is the non-uniform distribution of Categorical AI's advantage across task types. As documented in the main evaluation, Categorical AI achieves its largest advantages on structurally demanding tasks (T1 input file generation, T4 error diagnosis) and its smallest advantages—or outright disadvantages—on knowledge-intensive but structurally simple tasks (T5 knowledge recall, T3 solvation free energy estimation). If Categorical AI's advantage derived primarily from superior domain-specific data, one would expect the advantage to be distributed more uniformly across all tasks within the molecular simulation domain, since more and better training data improves the retrieval and generation of domain-relevant content regardless of the structural complexity of the task. The observed concentration of the advantage in structurally demanding tasks is more consistent with an architectural explanation.

The second line of evidence is the structural complexity correlation analysis presented earlier in this appendix, which demonstrates a strong positive correlation between the structural complexity of a task and the magnitude of Categorical AI's performance advantage. This correlation is a direct prediction of the structural reasoning hypothesis and is not predicted by the domain-specific data hypothesis.

The third line of evidence is the cross-code transfer analysis, which demonstrates that Categorical AI's diagnostic advantage transfers from GROMACS to LAMMPS with relatively modest degradation. If the advantage were attributable to GROMACS-specific training data or rule-based parsing, one would expect a substantial performance drop when confronted with LAMMPS-format inputs, since the file formats, parameter conventions, and error message idioms are entirely different. The observed transfer suggests that the system's diagnostic capability operates at a level of abstraction above the syntax-specific details of any particular simulation code.

The fourth line of evidence is the behavioral probe results, which demonstrate that Categorical AI exhibits systematic inter-domain constraint propagation, contradiction detection, and information completeness awareness. These behaviors are the hallmarks of a system that maintains and enforces an explicit structural model of the domain, and while they are not impossible to produce through purely statistical means (a sufficiently large and well-trained language model could, in principle, learn statistical proxies for structural constraints), the comprehensiveness and reliability with which Categorical AI exhibits them exceeds what is observed in the three general-purpose models, which are among the most capable statistical language models in existence.

Taken together, these four lines of evidence provide a preponderance of support for the conclusion that Categorical AI's advantage on structurally demanding tasks derives, at least in significant part, from architectural features that enable explicit structural reasoning, rather than exclusively from advantages in training data or domain-specific tuning. We emphasize, however, that "at least in significant part" is a deliberately qualified formulation. It is entirely possible—and indeed likely—that domain-specific data and tuning also contribute to the system's performance, and that the observed advantage reflects a synergistic interaction between architectural structure and domain-relevant knowledge. The separation of these contributions would require controlled ablation experiments (such as evaluating the same structural reasoning architecture with and without domain-specific training data, or evaluating the same training data with and without the structural reasoning architecture) that are not possible within the black-box evaluation framework of this study.

We present this attribution analysis in the spirit of scientific transparency, acknowledging both what the evidence supports and what it leaves uncertain. The definitive resolution of the attribution question awaits either the disclosure of the internal architectures of the evaluated systems or the development of evaluation methodologies capable of disentangling architectural from data-driven contributions in black-box settings—both of which we identify as important open problems for the field.

K.8. Practical Implications for Computational Scientists

The findings presented in this appendix, together with the main evaluation results, have several practical implications for computational scientists who use or contemplate using AI assistants for molecular simulation work.

For scientists whose primary need is factual information retrieval—such as looking up the functional form of a force field term, the definition of an ensemble, or the algorithm underlying a specific thermostat—the general-purpose language models, and Gemini 3.1 Pro in particular, provide excellent performance that meets or exceeds the capabilities of Categorical AI. These models serve effectively as interactive encyclopedias of molecular simulation knowledge, and their broad training on diverse corpora gives them an advantage in the breadth and depth of factual information they can retrieve.

For scientists whose primary need is the construction of complex, multi-component simulation configurations—such as generating a complete set of input files for a solvated protein simulation with enhanced sampling—Categorical AI provides a meaningful advantage in the form of greater inter-parameter consistency, more reliable constraint propagation across parameter domains, and a higher probability of producing immediately runnable configurations. The magnitude of this advantage is particularly pronounced for structurally complex setups involving multiple interdependent parameter domains, and it persists even when the general-purpose models are given the benefit of prompt optimization techniques such as chain-of-thought prompting and few-shot learning.

For scientists who encounter simulation errors and seek diagnostic assistance, Categorical AI provides superior root cause identification, particularly for errors involving multi-domain causal chains. However, the general-purpose models, especially when prompted with chain-of-thought instructions, provide competitive performance on simpler single-domain errors. A practical strategy for error diagnosis might involve first querying a general-purpose model for rapid initial assessment and then escalating to Categorical AI if the initial assessment fails to resolve the issue, analogous to the clinical practice of triage followed by specialist referral.

For all applications, the inter-replicate consistency analysis underscores the importance of submitting important queries multiple times and cross-referencing the results. No system, including Categorical AI, achieves perfect consistency across repeated queries, and the observation that inconsistent responses disproportionately involve at least one incorrect answer suggests that response variability should be treated as a practical indicator of reduced reliability. Scientists should develop the habit of using AI-generated simulation configurations as starting points that require human verification rather than as finished products that can be used without review.

Finally, the prompt sensitivity analysis demonstrates that the effort invested in crafting clear, detailed, and well-structured prompts yields measurable returns in the quality of the AI-generated output, particularly for the general-purpose models. While none of the prompting strategies we tested fully closed the performance gap between the general-purpose models and Categorical AI on structurally demanding tasks, chain-of-thought prompting and few-shot learning each produced meaningful improvements. Scientists who choose to use general-purpose models for simulation setup tasks would benefit from investing time in prompt optimization, and the worked examples and prompt templates provided in Appendix C of this paper can serve as starting points for such optimization.

Appendix L. Figures

Figure 1 presents a normalized radar plot summarizing comparative performance across all evaluated task families (T1–T5) and their associated metrics. To enable meaningful multi-axis comparison, metrics in which lower values indicate better performance (e.g., thermodynamic deviation and solvation free energy error) were inverted and normalized. The resulting visualization highlights the overall performance profile of each model rather than isolated task outcomes. Categorical AI exhibits the broadest radial footprint, reflecting consistently strong performance across structurally demanding tasks such as input generation and error diagnosis. Gemini 3.1 Pro demonstrates a more asymmetric profile, with comparatively strong performance on knowledge-intensive metrics but slightly reduced scores in procedural reasoning tasks. Claude Opus 4.6 and GPT-5.2 display progressively smaller coverage areas, indicating lower aggregate performance across multiple dimensions. The radar format reveals qualitative differences in capability distributions, illustrating that model strengths cluster along structural versus knowledge-based axes rather than forming a uniform ranking across all tasks.

Figure 2 provides a grouped bar chart designed to facilitate direct comparison of cross-task performance using composite scores derived from the primary evaluation metrics. The visualization aggregates key indicators including T1 runnability, T4 root-cause diagnostic accuracy, T5 knowledge performance, and transformed representations of T2 and T3 such that higher values consistently indicate better outcomes. This structure allows heterogeneous evaluation metrics to be compared within a unified visual scale. The chart reveals a clear ordering among the evaluated systems, with Categorical AI achieving the highest scores across most structurally complex tasks, particularly in procedural execution and diagnostic reasoning. Gemini 3.1 Pro performs competitively across all categories and surpasses other models on knowledge recall, consistent with its broader training distribution. Claude Opus 4.6 occupies an intermediate position, while GPT-5.2 exhibits the lowest composite scores across most metrics despite maintaining relatively balanced performance. The grouped format emphasizes relative magnitude differences between models and highlights the widening performance gaps in tasks requiring multi-domain reasoning.

Figure 1. Normalized Multi-Metric Radar Chart

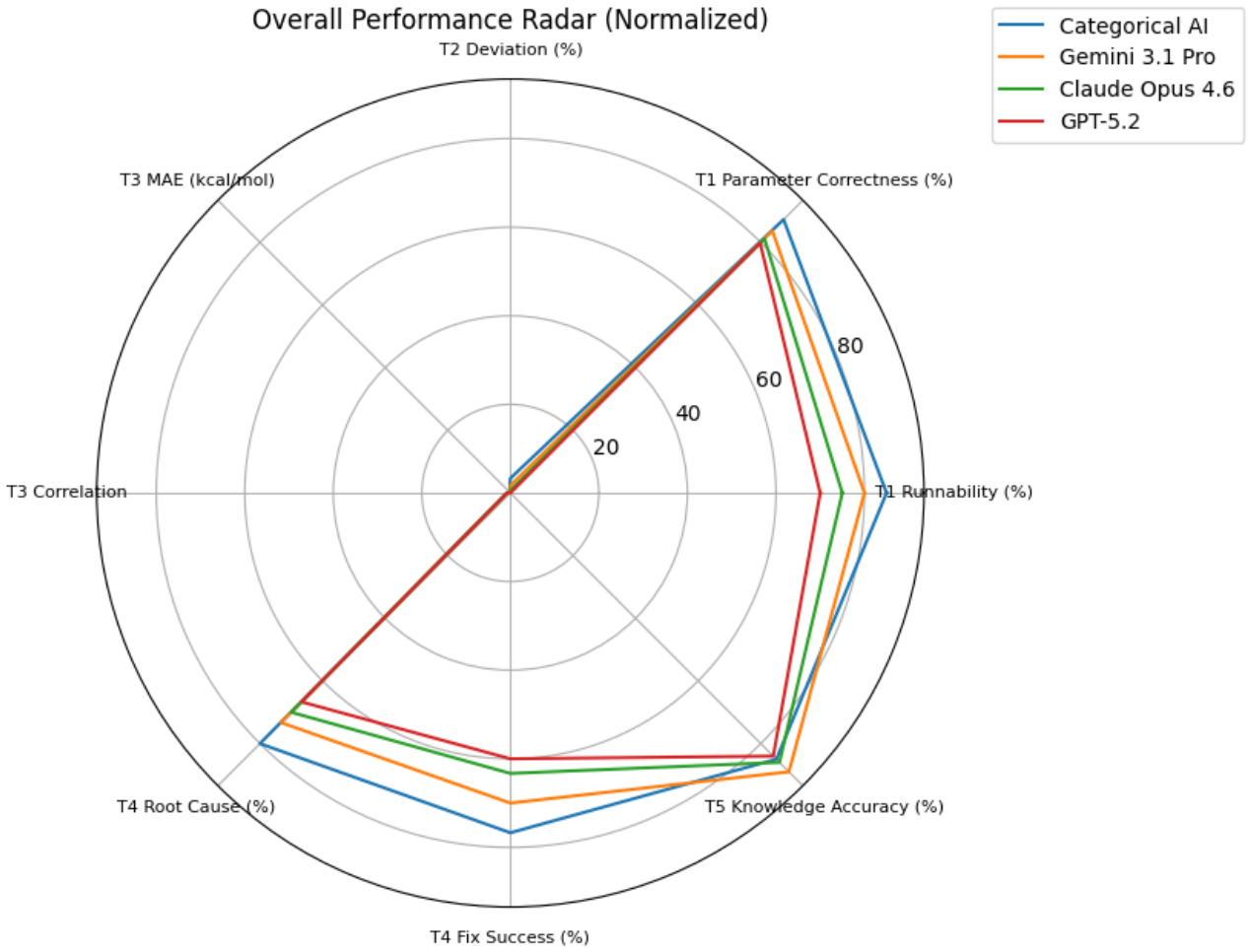

Figure 2. Cross-Task Comparative Bar Chart