Can Large Language Models Adequately Perform Symbolic Reasoning Over Time Series?

Anonymous ACL submission

Abstract

Uncovering hidden symbolic laws from time series data, as an aspiration dating back to Kepler's discovery of planetary motion, remains a core challenge in scientific discovery and artificial intelligence. While Large Language Models show promise in structured reasoning tasks, their ability to infer interpretable, context-aligned symbolic structures from time series data is still underexplored. To systematically evaluate this capability, we introduce **SymbolBench**, a comprehensive benchmark designed to assess symbolic reasoning over real-world time series across three tasks: *multivariate symbolic regression, Boolean network inference, and causal discovery*. Unlike prior efforts limited to simple algebraic equations, SymbolBench spans a diverse set of symbolic forms with varying complexity. We further propose a unified framework that integrates LLMs with genetic programming to form a closed-loop symbolic reasoning system, where LLMs act both as predictors and evaluators. Our empirical results reveal key strengths and limitations of current models, highlighting the importance of combining domain knowledge, context alignment, and reasoning structure to improve LLMs in automated scientific discovery.

1 Introduction

Centuries ago, Johannes Kepler revolutionized our understanding of the cosmos by discovering the laws of planetary motion (Gentner, 2002). Through meticulous analysis and rigorous reasoning over astronomical observations, captured as time series data of planetary positions, Kepler derived precise mathematical relationships that explained the complex, time-dependent dynamics of celestial orbits. Automating such a challenging process, where **hidden symbolic laws are extracted from time series**, is a long-standing aspiration in artificial intelligence (Reddy and Shojaee, 2025). However, achieving this automation presents fundamental reasoning-driven challenges, as time series data encapsulates dynamic behaviors and temporal dependencies that demand abstraction, generalization, and reasoning beyond mere pattern recognition to uncover underlying symbolic structures.

Recent advances in Large Language Models (LLMs) and Multimodal LLMs (MLLMs) show strong performance in complex reasoning tasks (Wang et al., 2024), but their ability to extract symbolic laws from time series remains underexplored. Traditional methods in symbolic regression, such as genetic programming (Makke and Chawla, 2024), often prioritize data fit at the expense of interpretability. Meanwhile, recent attempts (Merler et al., 2024; Li et al., 2024; Shojaee et al., 2024) to use LLMs in this domain have been limited; they employ them merely as domain-agnostic function generators, overlooking the models' core potential for deeper, theory-aligned reasoning. This shallow integration often results in proposed equations that lack contextual relevance. Additionally, most work focuses on algebraic expressions, neglecting other symbolic forms like logical formulas (Zhang et al., 2024) and causal relations (Assaad et al., 2022).

To address these gaps and provide further insights into the symbolic reasoning ability of LLMs for time series, a comprehensive benchmark is urgently required. Our work is guided by four key objectives: (a) **Real-world relevance:** To use real-world time series with ground-truth symbolic structures. (b) **Task difficulty:** To incorporate a diverse form of symbolic structures with varying complexity. (c) **Scale and balance:** To ensure a sufficiently large and balanced sample distribution across tasks. (d) **Unified framework:** To provide a unified framework to execute various tasks and establish connections with task-specific baselines.

To realize these objectives, we present three primary contributions.

(I) SymbolBench, a comprehensive benchmark designed to rigorously evaluate the symbolic reasoning capabilities of LLMs over time series with rich real-world contextual descriptions. It uniquely spans three core tasks covering major types of time series data: (a) **Multivariate symbolic regression** for continuous data to recover complex equations like coupled Ordinary Differential Equations (ODEs). (b) **Boolean network inference** (Zhang et al., 2024) for discrete systems to identify logical rules. (c) **Causal discovery** (Assaad et al., 2022) for multivariate data to uncover structured causal graphs. Each task includes challenging, real-world examples from domains such as biology, physics, and healthcare, with varying dimensionality and difficulty. To ensure both quality and coverage, we curate representative subsets from large databases, yielding a benchmark that is broader, more challenging, and more balanced than prior efforts.

(II) Unified Symbolic Reasoning Framework that enables the context-aware and knowledge-rich LLMs to play the dual role of predictors and judges in the process of hypothesis generation, testing, and refinement, while optionally including efficient genetic programming tools in a hybrid way.

(III) Critical empirical insights into the current strengths and limitations of LLMs and MLLMs in temporal symbolic reasoning. (a) LLMs surpass traditional baselines on multivariate symbolic regression and causal discovery, but fall short on Boolean network inference; (b) LLMs are able to perform a certain level of reasoning on the three tasks, with properly increasing test-time compute bringing moderate improvement. (c) Introducing context not only improves performance but also potentially guides the selection of generalizable symbolic structures. (d) Combining LLMs in complementary roles with genetic programming further boosts performance.

2 Related Work

Symbolic Expression Discovery. Symbolic regression (SR) aims to recover interpretable equations from time series data. Classical methods like Genetic Programming and sparse optimization (e.g., SINDy (Brunton et al., 2016), PySR (Cranmer, 2024)) prioritize accuracy and parsimony but face scalability challenges. Recent deep learning models (e.g., ODEformer (d'Ascoli et al., 2023), TPSR (Shojaee et al., 2023)) improve efficiency by treating SR as a translation task, though they require pretraining and lack iterative refinement. Related fields like Boolean network inference and causal discovery also seek symbolic structures from time series. More details are in Appendix A.

LLM Symbolic Reasoning. Due to the strong in-context learning and reasoning ability of LLMs that allow them to adapt to various tasks, they are able to perform logical inference(Ahn et al., 2024; Wang and Chen, 2023), and temporal symbolic reasoning (Fang et al., 2024). Despite the strong ability of LLMs, current research for SR tasks only applies them as function generators without a reasoning process. Though the generated functions may achieve a high fitting score, the reasoning process remains unknown to us and may fail to align with the context, with little real-world meaning. In addition, the current research focus has also skewed toward algebraic equations, with logical rules or causal relations remaining underexplored.

3 SymbolBench Dataset

To rigorously evaluate the reasoning abilities of LLMs on time series related science discovery, we introduce **SymbolBench**, a curated benchmark that aims to uncover symbolic structures from time series. The dataset spans diverse domains (e.g., physics, biology) and is structured around three core categories of symbolic structures. More details of the dataset are provided in Appendix B.

Coupled Differential Equations (CDEs). CDEs represent dynamic systems, yielding continuous and multivariate time series data. Let x_i denote the i-th variable of a multivariate time series, a coupled differential equation can be described as the following: $\frac{dx_i}{dt} = f_i(x_1, x_2, ..., x_n), i = 1, ..., n$, where the symbolic structures f_i describe dynamic interactions among state variables. The corresponding time series data consists of numerical solutions $\{x_i(t)\}$ over time, generated from initial conditions and system parameters. This setting reflects real-world dynamical systems in physics and engineering, where the complexity arises from variable interdependence. While the previous benchmark dataset, ODEbench (d'Ascoli et al., 2023), contains coupled ODEs from 1 dimension to 4 dimensions, the number of samples is small, and the class of dimensions is heavily imbalanced, with only three 4-dimensional ODEs. In this study, we further enrich ODEbench with more high-dimensional ODEs and provide a balanced dataset with over 156 samples. Each sample is accompanied by the variable descriptions and the domain name if available, as shown in Appendix B.

Boolean Networks (BNs). Multivariate time series with discrete values are also seen in the scientific domains. Derived from models used in systems biology, particularly in gene regulatory and signaling networks, Boolean networks represent each variable as a binary node whose state evolves according to logical expressions: $x_i^{(t+1)} = f_i(x_1^{(t)}, x_2^{(t)}, ..., x_n^{(t)}), x_i \in \{0, 1\}$, where f_i is a logical function composed of AND, OR, NOT, etc. The time series data is a sequence of binary vectors over discrete time steps, representing the dynamic evolution of the system. This setup emphasizes symbolic logic reasoning, state transitions, and rule discovery from temporal traces of binary states. In this study, we provide a curated subset of 65 Boolean networks from BioDivine (Pastva et al., 2023). For each sample, we provide a short description of the domain and the name of each variable.

Structured Causal Models (SCMs). Beyond specifying a specific data-generating process (e.g., via mathematical functions), studying causal dependencies among time series variables is valuable for uncovering interdependencies directly from raw data, akin to the broader task of causal discovery in temporal settings. This approach models systems using *Structural Causal Models* (SCMs), which can be expressed as a directed graph. For each variable x_i, the goal is to identify its parent variables x_j along with their corresponding time lags l, such that x_j at time $t - l$ causally influences x_i at time t: $x_i \leftarrow \{(x_j, l) \mid x_j \in X, ; l \in [1, M]\}$, where X denotes the set of all variables and M is the maximum considered lag. As the number of variables and potential lag intervals increases, the search space for an optimal SCM grows exponentially, making discovery more challenging. In this work, we extract SCMs from the CDEs in our curated dataset, as well as from additional CDEs in the Physiome database involving more than three variables, using functional analysis. Each sample is annotated with its corresponding SCM, resulting in 190 samples.

4 SymbolBench Reasoning Framework

Given an input time series with T time points $\{\mathbf{x}^i\}_{i=1}^T$, where $\mathbf{x}^i \in \mathbb{R}^D$ and D is the number of dimensions, our framework aims to generate a subset of symbolic structures for each time series. Figure 1 sketches the closed-loop workflow that combines LLMs into the automatic pipeline. The process starts with *Proposal Generation* (Sec. 4.1), which generates candidate expressions alongside their reasoning path if available. Then, the raw output is cleaned and used for *Verification* (Sec. 4.2), which assigns scores for each candidate. Candidates are then stored in a history pool. Finally, before the next round of generation, the *Context Manager* (Sec. 4.3) extracts candidates from the history pool based on certain rules and provides them as the context for the next round proposal generation. The loop repeats until either the stopping criterion is met or the budget exceeds the limit.

Table 1: Comparison of SymbolBench with existing benchmarks for LLMs across symbolic structures, evaluation setups, contextual data types, and reasoning.

Feature	LLM-SRBench	ODEBench	RealTCD	Ours
Symbolic Structures				
Scientific Eq.	128	63	–	156
Logical Exp.	–	–	–	65
SCM	–	–	2	190
Evaluation				
ID/OOD	✓	✓	–	✓
Reasoning	–	–	–	✓
Context				
Textual	✓	–	–	✓
Multimodal	–	–	–	✓

4.1 Proposal Generation

LLM-as-Predictor. To leverage the knowledge embedded in LLMs, recent studies have explored the direct generation of equations. In this work, we frame this approach as *LLM-as-Predictor*. To thoroughly assess both the capabilities of LLMs and the impact of iterative refinement, external context, and chain-of-thought reasoning, we introduce four distinct prompting strategies to evaluate the performance of LLMs/MLLMs comprehensively. **(a) Naive** prompt instructs the model to generate mathematical expressions without providing any contextual or historical information. **(b) Base** prompt builds upon the Naive prompt by incorporating previously generated expressions from the history pool, filtered by the context manager. **(b) Context** prompt further enhances the Base prompt by adding relevant contextual information, such as variable descriptions. **(c) CoT** prompt extends the Context prompt by enabling the model to perform step-by-step reasoning. The reasoning process is also recorded and subjected to qualitative verification. Details of all prompts are provided in Appendix I.

Hybrid Method. While the generation of symbolic structures is central to the pipeline, this part can also be addressed using operations from traditional genetic programming. In the context of CDEs and BNs, this involves applying operations like muta-

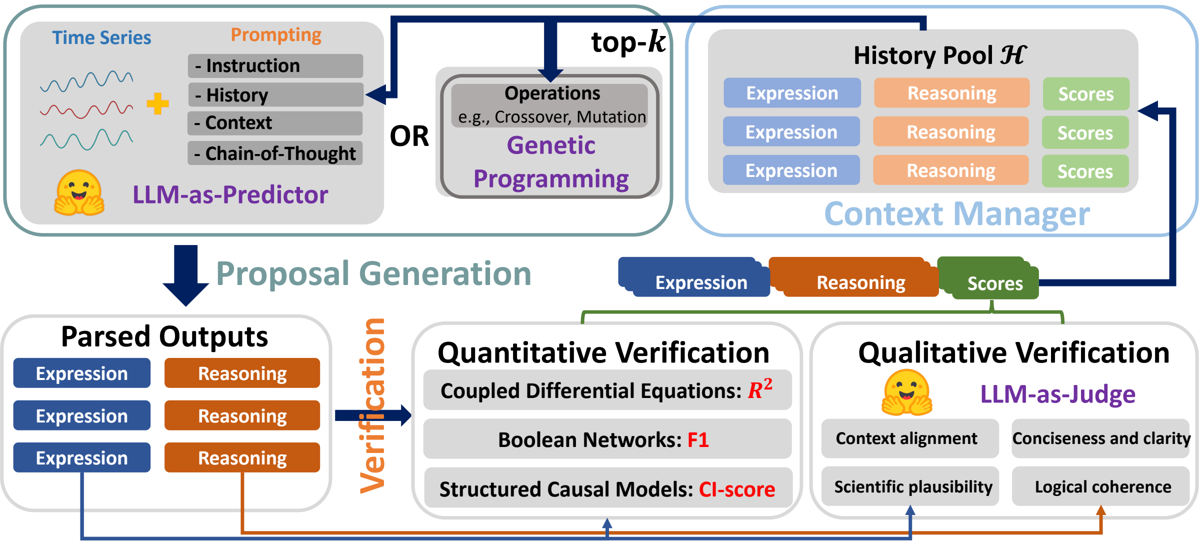


Figure 1: Iterative refinement framework. Candidate proposals are generated using either an LLM-as-Predictor or genetic programming operations. Each round of candidates undergoes quantitative and qualitative evaluation via validation tools and an LLM-as-Judge. Scored candidates are stored in a history pool, and a context manager decides contextual information for the next round.

tion and crossover on expression trees to produce new candidates. Since LLMs may still contribute to other components of the framework, e.g., verification, we phrase such a combination as the *Hybrid Method*, and further discuss it in Appendix F.

4.2 Verification

Verification in symbolic structure discovery assesses three key aspects: numerical fitness, symbolic fitness, and context alignment. While numerical fitness can be quantified, symbolic fitness and context alignment are harder to measure without ground truth. To address this, we introduce a dual-verification strategy combining standard quantitative metrics with rubric-based qualitative evaluation by *LLMs-as-Judges*.

4.2.1 Quantitative Verification

(a) For CDEs, we simulate time series data using the predicted functions and assess their numerical fitness using the averaged coefficient of determination: $R^2 = 1/D\sum_j^D(1 - \frac{\sum_{t=1}^T \|x_j^t - \hat{x}_j^t\|^2}{\sum_{t=1}^T \|x_j^t - \bar{\mathbf{x}}_\mathbf{j}\|^2})$, where x_j^t denotes the ground truth at time t for time series j, $\hat{x}_j^t$ the model prediction, and $\bar{\mathbf{x}}_j$ the empirical mean of the observed data. We also evaluate expression complexity based on symbolic structure, defined as the number of operations. **(b)** For Boolean networks, we simulate transitions from the predicted Boolean networks and evaluate predictive performance using the macro-averaged F1 score over T transitions: $\mathrm{F1} = \frac{1}{D*T}\sum_{j=1}^T\sum_{i=1}^D \frac{2\cdot\mathrm{TP}_i^j}{2\cdot\mathrm{TP}_i^j+\mathrm{FP}_i^j+\mathrm{FN}_i^j}$, where TP_i^j, FP_i^j, and FN_i^j are the true positives, false positives, and false negatives for node i at transition j, respectively. Expression complexity is computed analogously to CDEs. **(c)** For structural causal models, where the true data-generating process is typically unknown, we adopt a *Conditional Independence score (CI-score)* as a proxy for structural fidelity. This metric captures the strength of dependence between each child node and its parents, conditioned on the remaining parents. Let the parent set of X^i defined as $P_i = \{(\mathbf{x}_j, \ell) | j \in D, \ell \in \ell_{\max}\}$ where $\ell_{\max}$ is the maximum time lag. For each directed edge $(\mathbf{x}_j, \ell) \rightarrow i$ in the edge set E, we compute the partial correlation: $r_{j\rightarrow i} = \mathrm{Corr}\Big(X_j^{t-\ell}, X_i^t \;\Big|\; \Big\{x_k^{t-\ell'} : (\mathbf{x}_k, \ell') \in P_i, k \neq j\Big\}\Big)$. The CI-score for a candidate graph $\mathcal{G}$ is then given by: $\text{CI-score}(\mathcal{G}) = \frac{1}{|E|}\sum_{(\mathbf{x}_j,\ell)\rightarrow i\in E} |r_{j\rightarrow i}|$. A lower CI-score suggests stronger conditional independence and thus better structural plausibility.

4.2.2 Qualitative Verification

To assess the symbolic quality of the generated expressions, we employ a rubric-based scoring system ranging from 1 (poor) to 5 (excellent). Each qualitative score is assigned by an LLM acting as a judge (Hao et al., 2024), using a standardized evaluation rubric. The rubric consists of four core criteria: **(a) Context alignment:** Alignment with provided time series data and contextual descriptions; **(b) Scientific plausibility:** Alignment with possible physical laws or domain-specific constraints; **(c) Conciseness and clarity:** The readability and succinctness of the reasoning path; **(d) Logical coherence:** The consistency and step-by-step soundness of the derivation process.

4.3 Context Manager

The Context Manager is responsible for delivering relevant contextual information to the Proposal Generator during iterative refinement. Based on the chosen refinement strategy, the Context Manager selects and provides appropriate context for the next round of proposal generation. Throughout the process, the Context Manager maintains a *history pool* $\mathcal{H}$, which stores previously generated expressions, reasoning paths, and their associated verification scores in a structured DataFrame, while removing duplicates. In this study, we adopt a naive ranking-based strategy by only selecting the top-k highest-scoring candidates for further refinement.

5 Experiment

5.1 Experimental Setup

Baseline Models. For CDEs, we include the following baselines: PySR (Cranmer, 2024), a GP-based method; ProGED (Omejc et al., 2024), a probabilistic grammar-based approach; and ODEformer (d'Ascoli et al., 2023), a pretrained transformer-based model. For BNs, we evaluate LogicGep (Zhang et al., 2024), a GP-based method tailored for Boolean Network inference. For SCMs, we adopt several time-series causal discovery methods, including PCMCI (Runge et al., 2019), LPCMCI (Gerhardus and Runge, 2020), and j-PCMCI+ (Günther et al., 2023).

Evaluation Metrics. SymbolBench evaluates models on both in-distribution (ID) and out-of-distribution (OOD) data by applying new initial conditions. For **CDEs**, we report symbolic regression score: $SR^2 = \frac{1}{N}\sum_{i=1}^{N} R_i^2 \cdot \mathbb{I}(R_i^2 > 0)$, and accuracy for $R^2 > 0.9$: $ACC_{0.9}$. For **BNs**, we assess numerical similarity via precision, recall, F1, and bookmaker informedness: $recall + specificity - 1$. For **SCMs**, structural accuracy is measured using classification metrics and Structural Hamming Distance (SHD). Symbolic similarity (for CDEs and BNs) is evaluated via expression tree edit distance, and expression complexity via sympy's `count_ops` (Meurer et al., 2017). For BNs and SCMs, we also report accuracy for samples with F1 above a threshold: ACC_{thesh}.

5.2 Experimental Results

Across all three tasks, we select LLMs with various sizes and architectures, including Qwen2.5-14B (Team, 2024), Llama-3.2B (Dubey et al., 2024), Mathstral-7B (Jiang et al., 2023), GPT-4o-mini (Achiam et al., 2023), and ChatTS-14B (Xie et al., 2024). For GPT-4o-mini and ChatTS-14B, we use inputs with visual and temporal modalities. We present the results in Table 2, 3, and 4.

Obs. 1: LLMs demonstrate superior capability compared to baselines on CDEs and SCMs datasets, while failing to compete against baselines in Boolean network inference. For Boolean network inference, while LLMs consistently achieve positive bookmaker informedness scores, indicating performance better than random guessing, the genetic programming-based model, LogicGep, significantly outperforms all evaluated LLMs across nearly all evaluation metrics in both ID and OOD scenarios. This could be explained by both the symbolic and numerical fitting process. (a) *Symbolic fitting* Unlike CDEs, BN inference does not involve coefficient optimization. Compared with CDEs, even if the symbolic structure is not precisely correct, adjusting coefficients can still yield good numerical accuracy. Compared with SCMs, SCM inference focuses only on discovering causal relationships, which is a less stringent goal than recovering the full dynamics as in BN inference. (b) *Numerical fitting*: CDEs and SCMs are inferred from continuous time series, which form a consistent *chains of state transitions*, whereas BN inference uses state transitions with various initial conditions, forming *graphs of state transitions*, as shown in Appendix B and H.1, which can be hard to identify trends and summarize patterns.

Obs. 2: LLMs' performance degrades with problem difficulty. Across all models, performance consistently declines as the dimensionality of the system increases. While conventional methods such as PySR remain competitive in low-dimensional systems (e.g., dim = 1), baseline models tend to exhibit a steeper performance drop compared to LLMs. Similarly, there is a notable drop in accuracy when models are evaluated on OOD data, underscoring the increased complexity and generalization challenges posed by these scenarios.

Obs. 3: Chain-of-thought prompting does not consistently improve performance. Further introducing CoT prompting does not lead to consistent gains, especially on the CDEs dataset. An exception is ChatTS-14B, likely due to its specialized time series reasoning capabilities. While increasing test-time compute has been shown to improve outcomes in various tasks (Snell et al., 2024), and CoT can contribute to this effect, similar limitations have been observed in more complex tasks such as those in SciBench (Wang et al., 2023).

Obs. 4: Providing problem contexts improves LLM performance. Across all three tasks, compared to the Naive prompt, LLMs demonstrate the ability to leverage the provided context, resulting in higher numerical performance. In this setting, context functions as a form of conditioning, helping to constrain the solution space and guide the model toward more accurate and relevant inferences with a faster convergence rate, as shown in Appendix G.

5.3 Further Analysis

In this section, we explore the adaptability of our framework with genetic programming methods and the power of test-time compute. We formulate several key questions and takeaways as follows:

Q1: Do LLMs Employ Correct Symbolic Reasoning Paths? While previous research (Hao et al., 2024) has demonstrated that yielding correct answers does not necessarily yield correct reasoning paths, we find that general LLMs like Qwen2.5-14B with pure textual input are able to perform a certain level of reasoning (through CoT prompting) over the time series and the given context. As shown in Table H.2, the LLM is not only able to consider the meaning of each variable, but also the historical candidates.

Q2: How should test-time compute be structured for consistent improvement? While scaling test-time compute can enhance LLM performance (Snell et al., 2024), our findings indicate that the structure of this computation is critical for achieving consistent improvement. Simply increasing compute via naive CoT reasoning does not yield reliable gains, as shown in *Obs. 3*. This is because the depth of naive CoT reasoning often remains shallow and lacks key mechanisms such as reflection, verification, and backtracing that are present in more advanced **RLM** (Besta et al., 2025) with **Long CoT** (Chen et al., 2025), which involves more cognitive behaviors such as verification, reflection, and backtracing, etc. We use illustrative examples to demonstrate this in Appendix H.3.

Thus, to see consistent gains, test-time compute should be structured to facilitate a Long CoT. We explore two approaches: **First**, we analyze the effect of a Long CoT, generated by RLMs, within each reasoning epoch. This reveals that dedicating more compute to a longer, more detailed chain-of-thought leads to moderate but consistent improvements in the final prediction scores, as shown in Figure 3. **Second**, we view the entire iterative refinement process as a form of Long CoT. In this approach, the model verifies previous outputs and generates improved answers based on earlier, potentially flawed, solutions. As shown in Figure 2, the verification scores consistently improve as more refinement steps are added. From both approaches, we observe that the complexity of the best-fitting solutions increases with the amount of computation, echoing the human-like process of progressively constructing more sophisticated answers.

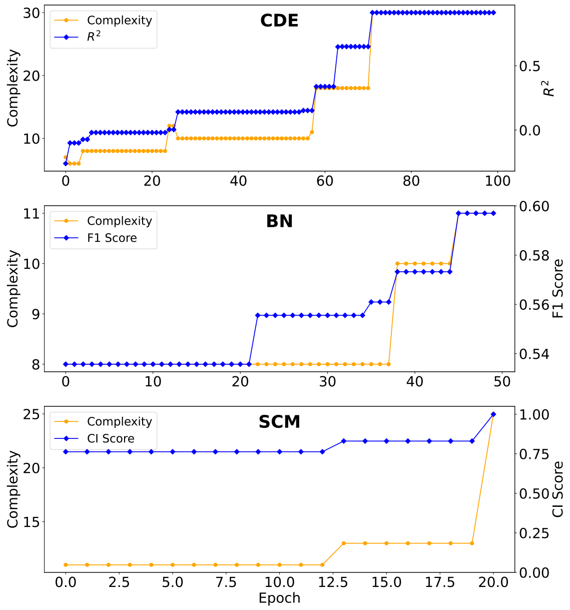


Figure 2: Evaluation scores improve with more iterations and test-time compute, as discussed in **Q2**.

Q3: Does the generalizability of prediction correlate with structural complexity? When faced with multiple symbolic structures that can fit the given time series, the conventional wisdom suggests that the simplest one is the most likely to be generalizable. Existing LLM-based approaches for symbolic regression (Shojaee et al., 2024; Li et al., 2024; Merler et al., 2024; Grayeli et al., 2024; Wang et al., 2025), rooted in the principle of Occam's Razor, advocate for choosing the expression with the least complexity. However, our experimental results challenge the assumption that simpler is always better. Our analysis shows that expressions with higher complexity can also achieve improved symbolic proximity and OOD performance in Table 2. This is because, in complex scientific do-

Table 2: Symbolic regression performance for CDEs across 4 dimensions. We use percentage for SR^2 and $ACC_{0.9}$. Yellow, Orange, and Cyan mark the first, second, and third place, respectively.

Dim		Metric	Baselines			Qwen2.5				Llama3.2		Mathstral		4o-text		4o-image		ChatTS	
			Pro.	PySR	ODE.	Naïve	Base	Ctx	CoT	Ctx	CoT	Ctx	CoT	Ctx	CoT	Ctx	CoT	Ctx	CoT
Dim=1		Complexity ↓	5.06	2.91	4.68	4.17	5.14	5.34	3.91	1.89	1.97	2.63	2.69	2.31	2.36	2.37	2.09	3.29	3.00
		Symbolic Prox. ↓	4.86	3.54	5.18	5.11	5.37	5.57	4.89	4.03	4.14	3.86	4.09	3.89	3.79	3.97	4.06	4.83	4.37
	ID	SR^2	95.50	96.90	83.40	93.30	97.20	99.00	97.40	95.10	94.90	96.20	96.20	92.80	95.40	95.70	95.30	95.10	97.00
		$ACC_{0.9}$	91.40	97.10	73.50	91.40	97.10	100.00	97.10	91.40	91.40	97.10	97.10	94.30	97.00	97.10	97.10	91.40	97.10
	OOD	SR^2	74.10	88.20	54.10	60.90	74.00	61.80	61.80	69.90	64.60	71.00	66.90	70.80	66.20	71.50	62.90	63.80	70.30
		$ACC_{0.9}$	65.70	85.70	47.10	45.70	63.60	52.90	51.40	51.40	48.60	62.90	60.00	54.30	51.50	54.30	45.70	51.40	54.30
Dim=2		Complexity ↓	5.71	7.00	9.05	8.76	9.75	10.00	9.07	6.93	7.07	7.93	8.35	6.91	6.64	6.33	6.33	9.88	9.63
		Symbolic Prox. ↓	10.90	8.21	9.98	11.20	11.60	11.40	11.10	10.00	9.86	10.30	10.40	10.40	9.98	10.10	10.00	11.70	11.40
	ID	SR^2	61.20	80.40	72.50	82.40	87.90	87.40	88.00	82.20	80.90	88.60	88.40	78.50	77.40	78.40	82.40	84.80	90.90
		$ACC_{0.9}$	42.90	76.90	47.60	71.10	77.30	80.00	76.70	65.10	65.10	79.10	79.10	62.20	59.50	62.80	62.80	76.70	83.70
	OOD	SR^2	18.80	54.00	34.60	44.90	52.90	45.30	49.70	40.60	47.30	46.10	46.80	42.70	40.20	38.10	45.60	49.50	44.70
		$ACC_{0.9}$	12.50	42.10	18.60	31.00	44.40	34.10	35.90	32.60	34.90	34.90	33.30	29.50	22.50	23.80	29.30	43.60	33.30
Dim=3		Complexity ↓	7.87	8.82	13.90	16.60	17.40	18.30	15.50	15.70	17.30	16.30	15.40	12.60	12.30	12.00	11.50	17.00	17.20
		Symbolic Prox. ↓	30.30	25.70	30.50	29.80	30.10	29.70	28.60	28.80	30.00	29.50	29.00	29.50	27.40	29.20	29.20	29.40	29.60
	ID	SR^2	20.90	67.10	36.30	71.30	74.30	75.30	70.90	62.10	57.40	65.20	64.60	70.00	63.70	65.30	66.90	70.80	68.40
		$ACC_{0.9}$	10.90	60.50	18.40	61.20	63.30	67.30	53.10	49.00	43.50	57.10	57.10	56.30	50.00	49.00	53.10	59.20	57.10
	OOD	SR^2	8.60	49.90	30.00	47.20	43.20	44.30	40.90	43.10	35.90	38.70	45.00	44.70	39.40	39.20	37.30	40.20	42.40
		$ACC_{0.9}$	4.50	36.10	15.20	39.50	38.60	36.60	33.30	34.00	28.60	33.30	40.90	38.30	31.70	31.10	29.20	32.60	33.30
Dim=4		Complexity ↓	9.70	10.60	14.40	22.90	23.20	23.10	18.50	20.60	19.60	17.90	18.70	14.50	13.90	13.90	13.60	22.60	22.10
		Symbolic Prox. ↓	32.70	30.30	36.40	35.20	35.40	34.10	32.90	34.70	32.70	32.50	32.40	34.80	32.90	32.20	31.90	38.00	36.30
	ID	SR^2	11.50	74.60	34.80	92.70	93.60	93.30	91.00	49.40	61.10	78.00	78.70	88.60	86.90	86.10	89.10	78.70	80.00
		$ACC_{0.9}$	8.70	65.20	29.60	87.90	94.10	93.90	89.30	40.70	52.00	74.10	75.00	78.80	70.00	71.40	85.70	74.10	80.80
	OOD	SR^2	7.00	23.80	11.40	29.20	19.10	19.40	34.60	28.30	21.30	30.70	36.90	26.90	27.60	30.40	30.50	25.00	24.50
		$ACC_{0.9}$	0.00	21.70	7.10	19.40	15.60	12.50	33.30	12.00	13.60	24.00	30.80	19.40	20.70	15.40	21.40	21.70	17.40

Table 3: Comparison of Boolean network inference across ID and OOD settings.

Model	**Setting**	**ID**							**OOD**							**Symb. Prox. ↓**	**Comp. ↓**
		Prec.	**Rec.**	**Acc.**	**B.I.**	**$ACC_{0.5}$**	**$ACC_{0.7}$**	**$ACC_{0.8}$**	**Prec.**	**Rec.**	**Acc.**	**B.I.**	**$ACC_{0.5}$**	**$ACC_{0.7}$**	**$ACC_{0.8}$**		
LogicGep	∼	93.6	92.7	95.2	88.7	98.5	98.5	96.9	84.7	86.5	89.5	76.5	98.5	86.2	76.9	12.39	12.39
Qwen2.5-14B	Naïve	58.7	71.7	66.5	29.5	86.2	33.8	6.2	56.4	71.3	64.5	26.6	80.0	24.6	7.7	12.87	14.84
	Base	54.7	73.3	63.5	27.3	80.0	20.0	3.1	53.8	72.8	62.5	25.9	83.1	16.9	3.1	12.33	16.73
	Context	57.8	77.2	67.1	29.1	87.7	38.5	9.2	56.1	77.2	65.5	27.0	87.7	30.8	10.8	12.39	16.86
	CoT	58.3	73.9	65.4	30.3	92.3	27.7	4.6	56.2	73.9	63.2	27.9	84.6	24.6	3.1	11.96	14.79
Llama3.2-3B	Context	51.6	74.8	62.4	24.5	60.0	18.5	1.5	47.4	71.1	58.2	16.9	52.3	9.2	1.5	14.12	17.67
	CoT	52.0	73.1	62.3	23.0	61.5	20.0	1.5	49.9	67.8	58.3	15.7	56.9	13.8	1.5	14.30	19.59
Mathstral-7B	Context	49.5	73.3	60.3	19.2	50.8	15.4	1.5	48.0	72.2	58.0	16.4	53.8	10.8	1.5	12.74	20.14
	CoT	51.3	74.4	62.6	23.9	61.5	21.5	3.1	48.3	71.7	58.3	16.9	60.0	16.9	3.1	12.16	20.48
GPT-4o-mini	Context	51.6	58.9	62.6	21.2	47.7	15.4	0.0	51.3	58.5	61.4	20.0	53.8	13.8	0.0	13.24	30.07
	CoT	52.2	58.9	63.5	23.5	53.8	4.6	0.0	51.7	60.6	62.2	23.0	53.8	7.7	0.0	11.20	32.42

mains such as biology or physics, the ground-truth expression may inherently be more structurally complex while being numerically simpler in the ID setting. Imposing a strong simplicity criterion can hinder the discovery or selection of the more generalizable *context-aligned* candidates. As detailed in Appendix E and Table 5, our analysis suggests that: *Candidate ranking and selection should not solely rely on complexity and may benefit from considering contextual information*.

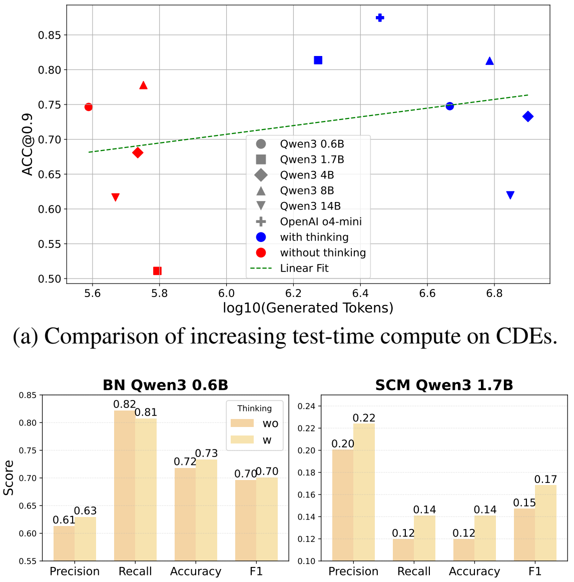
Q4: Can LLMs and genetic programming (GP) be combined for enhanced performance? While LLMs show superior performance on CDE and SCM inference, GP holds value for its cost, control over complexity, and stronger performance in BN inference. To capitalize on the advantages of both, we explored two primary methods for creating a high-performance hybrid system, as detailed in Appendix F. The key is to use the contextual understanding of LLMs to "inject context and knowledge" into the GP workflow, boosting its performance. This can be achieved in two main ways: (a) *LLM-Guided Initialization*: Use an "LLM-as-Predictor" to generate a high-quality initial population of candidate expressions for the GP to evolve. This provides the GP with a context-aware and promising starting point. (b) *LLM-Guided Evaluation*: Employ an "LLM-as-Judge" to provide context-enhanced evaluation scores during the GP's fitness assessment. This helps guide the evolution-

Table 4: Performance comparison of causal discovery methods and LLM-based approaches.

Model	Setting	F1	Prec.	Recall	FDR↓	$\text{ACC}_{0.5}$	$\text{ACC}_{0.7}$	$\text{ACC}_{0.8}$	SHD↓	Complx↓
PCMCI	∼	52.7	52.2	63.7	46.1	55.3	23.7	7.9	95.28	96.76
LPCMCI	∼	52.0	68.6	45.9	29.1	56.3	18.4	5.3	25.35	14.54
j-PCMCI+	∼	46.2	58.1	43.3	39.1	45.3	13.2	7.4	49.36	38.83
Qwen2.5-14B	Naïve	49.7	59.4	45.0	40.6	43.7	17.9	8.4	35.54	18.93
	Base	50.5	61.4	45.0	38.6	47.4	18.4	8.9	34.58	18.10
	Context	53.4	63.0	48.3	37.0	53.7	19.5	7.9	31.69	18.61
	CoT	51.3	61.9	46.3	38.1	52.6	20.0	8.4	39.36	21.76
Llama3.2-3B	Context	51.3	56.1	49.0	43.9	37.4	13.7	8.9	41.89	25.43
	CoT	51.8	56.4	49.5	43.6	48.4	22.6	11.6	44.40	25.75
Mathstral-7B	Context	47.2	54.8	43.1	45.2	40.5	15.8	7.9	37.70	20.48
	CoT	49.8	58.0	45.4	42.0	44.7	14.2	5.8	34.30	19.38
GPT-4o-mini	Context	36.9	59.9	27.6	40.1	24.7	5.3	1.6	44.29	13.36
	CoT	37.1	57.5	28.5	42.5	24.2	3.2	2.6	38.90	13.08
ChatTS-14B	Context	54.1	72.3	46.2	27.7	58.9	22.6	11.6	32.48	15.07
	CoT	54.4	72.3	46.6	27.7	61.1	25.3	10.5	31.10	14.70

Table 5: Correlations between complexity and OOD $ACC_{0.9}$ across four dimensions with and without context in symbolic regression task.

Condition	Dim 1	Dim 2	Dim 3	Dim 4
w/o context	−0.672	−0.384	−0.425	0.584
w/ context	0.097	0.728	0.165	−0.174



(a) Comparison of increasing test-time compute on CDEs.

(b) Comparison of w and w/o thinking on BNs and SCMs.

Figure 3: Comparison of performance with and without thinking.

ary search toward solutions that are not only numerically accurate but also scientifically plausible. As shown in Tables 6 and 7, combining LLMs with GP in these roles demonstrably improves performance.

6 Conclusion and Outlook

We introduce **SymbolBench**, a real-world benchmark for symbolic structure discovery, and a **Unified Symbolic Reasoning Framework** that enables LLMs (optionally with GP) to generate and judge hypotheses across tasks. Experiments show: (i) LLMs beat baselines on multivariate symbolic regression and causal discovery but lag on Boolean network inference; (ii) more test-time compute yields only modest gains; (iii) contextual grounding boosts accuracy and generalizability; and (iv) LLM–GP hybrids further improve performance.

Future Opportunities. Based on our observations and takeaways, we summarize some potential future directions as follows: (i) task-specific scaling of test-time compute and reasoning depth; (ii) richer, knowledge-heavy context to guide hypotheses; (iii) context-aware criteria beyond syntactic simplicity (e.g., plausibility, robustness, causal faithfulness); and (iv) broader symbolic targets plus verifiable reasoning traces for interpretability.

Table 6: Hybrid method on CDEs. We use the same GPT-4o-mini as judge and predictor.

Method	SR^2_{ID}	SR^2_{OOD}	Sym. Prox.↓	Comp.↓
GPLearn	27.6	14.9	6.083	1.333
GPT-4o-mini	39.3	24.1	7.438	7.406
GPLearn + LLM-as-Judge	31.7	16.3	6.550	1.200
LLM-as-Predictor + GPLearn	89.5	69.3	5.045	2.682

Table 7: Hybrid method on BNs. We use the same GPT-4o-mini as judge and predictor.

Method	Setting	Metrics					
		Prec.	Recall	F1	Acc	BM	Comp.↓
LogicGep	ID	88.0	88.6	88.0	91.7	80.9	0.773
	OOD	78.9	83.0	80.0	85.6	69.2	0.773
GPT-4o-mini	ID	52.8	65.6	57.8	61.2	23.6	2.422
	OOD	49.9	62.3	54.5	58.3	17.3	2.422
LogicGep + Judge	ID	89.3	92.8	90.9	93.2	85.6	1.012
	OOD	78.6	85.2	80.4	85.8	71.0	1.012
Predictor + LogicGep	ID	72.2	84.7	77.5	80.6	61.0	0.993
	OOD	63.1	77.2	67.9	73.2	46.9	0.993

7 Limitations

This work has several limitations stemming from computational constraints. First, we did not evaluate larger open-source models (e.g., DeepSeek-R1). Second, we capped the budget at 100 generation epochs and a maximum of 20 retries per epoch for each run; samples without candidates that could reach the tolerance will be rerun once again to mitigate uncertainty. Finally, we examined only one LLM and one GP in the hybrid architecture, and did not analyze the effects of reasoning strategies or model size.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Charles K Assaad, Emilie Devijver, and Eric Gaussier. 2022. Survey and evaluation of causal discovery methods for time series. *Journal of Artificial Intelligence Research*, 73:767–819.

Maciej Besta, Julia Barth, Eric Schreiber, Ales Kubicek, Afonso Catarino, Robert Gerstenberger, Piotr Nyczyk, Patrick Iff, Yueling Li, Sam Houliston, and 1 others. 2025. Reasoning language models: A blueprint. *arXiv preprint arXiv:2501.11223*.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. 2016. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*.

Miles Cranmer. 2024. Pysr: High-performance symbolic regression in python and julia. *Astrophysics Source Code Library*, pages ascl–2409.

Stéphane d'Ascoli, Sören Becker, Alexander Mathis, Philippe Schwaller, and Niki Kilbertus. 2023. Odeformer: Symbolic regression of dynamical systems with transformers. *arXiv preprint arXiv:2310.05573*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Meng Fang, Shilong Deng, Yudi Zhang, Zijing Shi, Ling Chen, Mykola Pechenizkiy, and Jun Wang. 2024. Large language models are neurosymbolic reasoners. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17985–17993.

Shuhua Gao, Changkai Sun, Cheng Xiang, Kairong Qin, and Tong Heng Lee. 2022. Learning Asynchronous Boolean Networks From Single-Cell Data Using Multiobjective Cooperative Genetic Programming. *IEEE Transactions on Cybernetics*, 52(5):2916–2930.

Dedre Gentner. 2002. Analogy in scientific discovery: The case of johannes kepler. In *Model-based reasoning: Science, technology, values*, pages 21–39. Springer.

Andreas Gerhardus and Jakob Runge. 2020. High-recall causal discovery for autocorrelated time series with latent confounders. *Advances in neural information processing systems*, 33:12615–12625.

Arya Grayeli, Atharva Sehgal, Omar Costilla Reyes, Miles Cranmer, and Swarat Chaudhuri. 2024. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37:44678–44709.

Wiebke Günther, Urmi Ninad, and Jakob Runge. 2023. Causal discovery for time series from multiple datasets with latent contexts. In *Uncertainty in Artificial Intelligence*, pages 766–776. PMLR.

Shibo Hao, Yi Gu, Haotian Luo, Tianyang Liu, Xiyan Shao, Xinyuan Wang, Shuhua Xie, Haodi Ma, Adithya Samavedhi, Qiyue Gao, and 1 others. 2024. Llm reasoners: New evaluation, library, and analysis of step-by-step reasoning with large language models. *arXiv preprint arXiv:2404.05221*.

Uzma Hasan, Emam Hossain, and Md Osman Gani. 2023. A survey on causal discovery methods for iid and time series data. *arXiv preprint arXiv:2303.15027*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Yanjie Li, Weijun Li, Lina Yu, Min Wu, Jingyi Liu, Wenqiang Li, Shu Wei, and Yusong Deng. 2024. Mllm-sr: Conversational symbolic regression base multi-modal large language models. *arXiv preprint arXiv:2406.05410*.

Zekun Li, Shiyang Li, and Xifeng Yan. 2023. Time series as images: Vision transformer for irregularly sampled time series. *Advances in Neural Information Processing Systems*, 36:49187–49204.

Nour Makke and Sanjay Chawla. 2024. Interpretable scientific discovery with symbolic regression: a review. *Artificial Intelligence Review*, 57(1):2.

Matteo Merler, Katsiaryna Haitsiukevich, Nicola Dainese, and Pekka Marttinen. 2024. In-context symbolic regression: Leveraging large language models for function discovery. *arXiv preprint arXiv:2404.19094*.

Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, and 8 others. 2017. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103.

T Nathan Mundhenk, Mikel Landajuela, Ruben Glatt, Claudio P Santiago, Daniel M Faissol, and Brenden K Petersen. 2021. Symbolic regression via neural-guided genetic programming population seeding. *arXiv preprint arXiv:2111.00053*.

Nina Omejc, Boštjan Gec, Jure Brence, Ljupčo Todorovski, and Sašo Džeroski. 2024. Probabilistic grammars for modeling dynamical systems from coarse, noisy, and partial data. *Machine learning*, 113(10):7689–7721.

Samuel Pastva, David Šafránek, Nikola Beneš, Luboš Brim, and Thomas Henzinger. 2023. Repository of logically consistent real-world boolean network models. *bioRxiv*, pages 2023–06.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Chandan K Reddy and Parshin Shojaee. 2025. Towards scientific discovery with generative ai: Progress, opportunities, and challenges. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 28601–28609.

Jakob Runge, Ewen Gillies, Eric V Strobl, and Shay Palachy-Affek. 2022. Tigramite–causal inference and causal discovery for time series datasets. Available at https://github.com/jakobrunge/tigramite?tab=readme-ov-file.

Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. 2019. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science advances*, 5(11):eaau4996.

Parshin Shojaee, Kazem Meidani, Amir Barati Farimani, and Chandan Reddy. 2023. Transformer-based planning for symbolic regression. *Advances in Neural Information Processing Systems*, 36:45907–45919.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K Reddy. 2024. Llm-sr: Scientific equation discovery via programming with large language models. *arXiv preprint arXiv:2404.18400*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Jianxun Wang and Yixiang Chen. 2023. A review on code generation with llms: Application and evaluation. In *2023 IEEE International Conference on Medical Artificial Intelligence (MedAI)*, pages 284–289. IEEE.

Runxiang Wang, Boxiao Wang, Kai Li, Yifan Zhang, and Jian Cheng. 2025. Drsr: Llm based scientific equation discovery with dual reasoning from data and experience. *arXiv preprint arXiv:2506.04282*.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. 2023. Scibench: Evaluating college-level scientific problem-solving abilities of large language models. *arXiv preprint arXiv:2307.10635*.

Yiqi Wang, Wentao Chen, Xiaotian Han, Xudong Lin, Haiteng Zhao, Yongfei Liu, Bohan Zhai, Jianbo Yuan, Quanzeng You, and Hongxia Yang. 2024. Exploring the reasoning abilities of multimodal large language models (mllms): A comprehensive survey on emerging trends in multimodal reasoning. *arXiv preprint arXiv:2401.06805*.

Zhe Xie, Zeyan Li, Xiao He, Longlong Xu, Xidao Wen, Tieying Zhang, Jianjun Chen, Rui Shi, and Dan Pei. 2024. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *arXiv preprint arXiv:2412.03104*.

Dezhen Zhang, Shuhua Gao, Zhi-Ping Liu, and Rui Gao. 2024. Logicgep: Boolean networks inference using symbolic regression from time-series transcriptomic profiling data. *Briefings in Bioinformatics*, 25(4):bbae286.

Haochuan Zhang, Chunhua Yang, Jie Han, Liyang Qin, and Xiaoli Wang. 2025. Tempogpt: Enhancing temporal reasoning via quantizing embedding. *arXiv preprint arXiv:2501.07335*.

Appendices

A Related Work

symbolic structure Discovery. Discovering symbolic laws from time series data is a central objective in many scientific discovery tasks. One prominent approach is Symbolic Regression, which seeks closed-form expressions that accurately model the observed data. Classical methods such as Genetic Programming (GP) (Mundhenk et al., 2021) are powerful but computationally intensive and often sensitive to the choice of operators and fitness functions. Alternatively, sparse optimization techniques like SINDy (Brunton et al., 2016) and PySR (Cranmer, 2024) aim to identify parsimonious models by leveraging sparsity in the function space. More recently, deep learning-based models have been introduced to enhance efficiency and scalability. Methods such as ODEformer (d'Ascoli et al., 2023) and TPSR (Shojaee et al., 2023) reformulate symbolic discovery as a sequence-to-sequence translation task, mapping time series data to symbolic equations. These approaches generate high-quality expressions and offer improved computational efficiency. However, they typically require large-scale pretraining and often lack the capability for iterative refinement and adaptation across diverse scientific domains. Beyond symbolic regression, related tasks such as Boolean Network Inference(Zhang et al., 2024) and Causal Discovery(Hasan et al., 2023) also aim to extract symbolic structures from time series data. These methods seek to uncover underlying logical or causal relationships, further emphasizing the broader interest in interpretable, symbolic representations of dynamical systems.

LLM Symbolic Reasoning. Due to the strong in-context learning and reasoning ability of Large language models (LLMs) that allow them to adapt to various tasks, they are able to perform logical inference(Ahn et al., 2024; Wang and Chen, 2023), and temporal symbolic reasoning (Fang et al., 2024). Despite the strong ability of LLMs, current research for SR tasks only applies them as proposal generators without a reasoning process. Though the generated functions may achieve a high fitting score, the reasoning process remains unknown to us and may fail to align with the context, with little real-world meaning. In addition, the current research focus has also skewed toward algebraic equations, with little work on logic rules or causal relations. Our SymbolBench, compared to previous benchmarks as shown in Table 1, addresses this by systematically evaluating both LLMs and MLLMs on SR for time series data, covering varied symbolic forms and emphasizing the transparency of the reasoning process. In addition, compared to prior benchmarks, we provide a more comprehensive evaluation including In-distribution and Out-of-distribution settings over coupled ODEs, logical expressions, and structured causal models.

B SymbolBench Dataset

SymbolBench uses three different datasets to evaluate the ability of LLMs/MLLMs to uncover the symbolic laws from time series data. We include variable description, domain name, and the time series trajectory as additional context. Examples of the dataset are shown in Table 8.

In addition, we provide a more detailed illustration of the verification process of the three tasks as follows:

(a) **CDEs:** We use an LLM to generate the skeleton of the continuous-time dynamical system, replacing each unknown coefficient with a placeholder "c". An optimizer then fits these coefficients to the training time series. To avoid the high cost of repeatedly solving and differentiating through a full ODE solver, we adopt the finite–difference approximation strategy from ODEFormer (d'Ascoli et al., 2023). Once the functional form $\hat{f}(\mathbf{x};\boldsymbol{\phi})$ is obtained, where ϕ is the fitted coefficients, we generate the final numerical solution using SciPy's `scipy.integrate.solve_ivp`, and compare the trajectory to the ground truth:

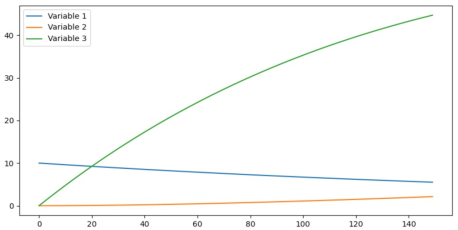
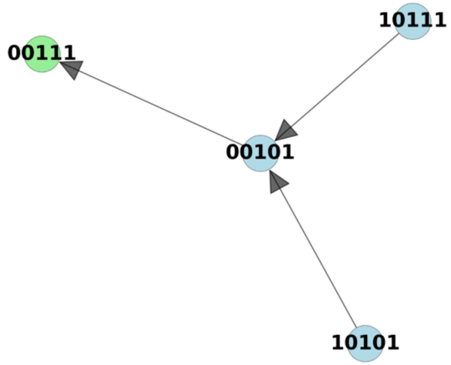
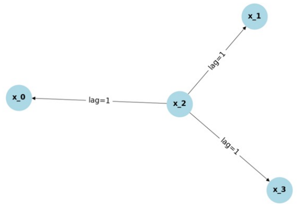
$$\texttt{solve\_ivp}\big(\hat{f}(\mathbf{x};\boldsymbol{\phi}),\ \mathbf{x}(t_0),\ t_0,\ldots,t_n,\texttt{method=LSODA}\big). \tag{1}$$

The distribution of sample dims is shown in Figure 4a.

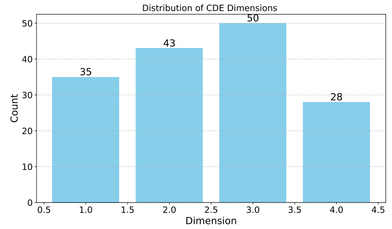
(b) **BNs:** For Boolean Networks, the LLM directly outputs a set of logical update rules (e.g. $x_i(t+1) = x_j(t) \wedge \neg x_k(t)$). Since there are no continuous parameters to fit, we simply simulate the network from the known initial state $x(t_0)$ and compute the F1 score over all bits and time steps to assess agreement with the true dynamics. The distribution of sample dims is shown in Figure 4b.

(c) **SCMs:** For Structured Causal Models, the LLM predicts the possible causal relations among all variables, forming a directed graph. Since the predicted SCMs can not directly pro-

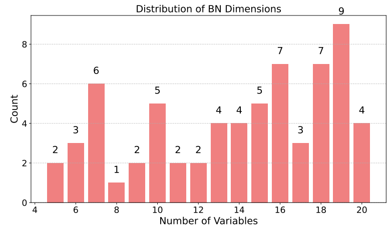
Table 8: Summary of datasets, ground truths, and variable context with trajectory images.

Dataset Example	GroundTruth	Variable Description	Domain	Trajectory
Coupled Differential Equation	dx1/dt = (c0 + c1 / (1 + c2 * x2)) - c3 * x1	x1: ng_ml	Endocrine	
	dx2/dt = (c8 + c9 * x3) - c10 * x2	x2: microg_dl		
	dx3/dt = (c4 + c5 * x1) / (1 + c6 * x2) - c7 * x3	x3: pg_ml		
Boolean Network	x1 = (NOT (x3 OR x5) OR NOT (x5 OR x3))	x1: v_Coup_fti	Cortical Area Development	
	x2 = (x1 AND NOT ((x3 OR x5) OR x4))	x2: v_Emx2		
	x3 = ((x3 AND x5) AND NOT x2)	x3: v_Fgf8		
	x4 = (x5 AND NOT (x2 OR x1))	x4: v_Pax6		
	x5 = (x3 AND NOT x2)	x5: v_Sp8		
Structured Causal Model	∼	x0: membrane (millivolt)	Calcium Dynamics	
	∼	x1: rapidly_activating_K_current_n_gate (dimensionless)		
	∼	x2: Ca_i in component ionic_concentrations (micromolar)		
	∼	x3: None		

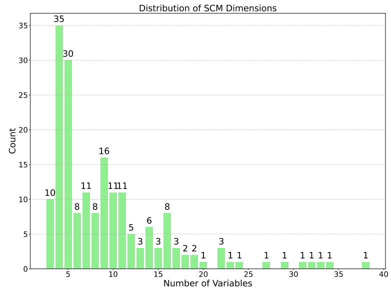
duce numerical solutions, to quantify how well the predicted SCM explains the data, we compute the sample partial correlation between x_i and each candidate parent in $\mathrm{pa}(x_i)$ (conditioning on the parents), following the protocol of Runge et al. (Runge et al., 2019). The final score for each node is the mean of its absolute partial correlations, and we average over all nodes to obtain the overall SCM score. The distribution of sample dims is shown in Figure 4c.



(a) Dimension distribution of Coupled Differential Equations used in SymbolBench.



(b) Dimension distribution of Boolean Networks used in SymbolBench.



(c) Dimension distribution of Structured Causal Models used in SymbolBench.

Textual Context. For regular LLMs that only accept textual inputs, all inputs to LLMs are formatted as structured textual prompts. Time series data are serialized into strings, supplemented with contextual metadata such as domain information, variable meanings, and prior scored expressions. This enables language models to reason over both data patterns and contextual priors.

Visual and Temporal Context. Multimodal LLMs have shown promise in handling visual and temporal data. While visual inputs are traditionally used in vision-language tasks (Radford et al., 2021), recent research has extended MLLMs to time series domains (e.g., forecasting via visual encodings (Li et al., 2023)). Recent models like ChatTS (Xie et al., 2024) and TempoGPT (Zhang et al., 2025) enable joint reasoning over temporal and textual modalities. In SymbolBench, we explore the use of MLLMs to incorporate both visual time series plots and encoded temporal embeddings.

C LLMs for Benchmarking

We evaluate six representative LLMs, chosen for their diversity in size, training data, and specialization:

(a) **Qwen Series (Team, 2024, 2025):** Qwen2.5-14B (Team, 2024) is a 14.7-billion-parameter causal Transformer (13.1 B non-embedding) built on RoPE, SwiGLU, RMSNorm, and QKV-bias that extends context support to 128 K tokens (with generation up to 8 K) and delivers significantly richer knowledge, advanced coding, and mathematical reasoning (via domain-expert submodels), robust instruction following, long-form text, and structured output (e.g., tables, JSON), and out-of-the-box multilingual fluency across 29+ languages. Following Qwen2.5-14B, Qwen3 (Team, 2025) series is the newest generation in the Qwen family, combining dense and Mixture-of-Experts architectures to deliver seamless mode-switching—"thinking" for deep logical reasoning, math, and coding, and "non-thinking" for fast, general dialogue.

(b) **Llama 3.2-3B (Dubey et al., 2024):** Llama 3.2 is a family of multilingual 1 B and 3 B–parameter pretrained and instruction-tuned text-in/text-out models, with its instruction-tuned versions specially optimized for dialogue, agentic retrieval, and summarization across dozens of languages—consistently outperforming many open-source and proprietary chat models on standard industry benchmarks.

(c) **Mathstral-7B (Jiang et al., 2023):** Mathstral is a 7 billion-parameter LLM released by Mistral AI as a tribute to Archimedes' 2311th anniversary, built on Mistral 7B with a 32K token context window and fine-tuned for advanced multi-step mathematical and scientific reasoning. Developed in collaboration with Project Numina, it achieves state-of-the-art performance for its size.

(d) **ChatTS-14B (Xie et al., 2024):** ChatTS-

14B is a multimodal LLM explicitly designed around time series as its core modality, offering native support for multivariate sequences of varying lengths and dimensions, preserving raw numerical fidelity for precise statistical queries, and enabling interactive, conversational exploration and reasoning over time-series data—while also integrating seamlessly into existing LLM workflows (including vLLM) with provided code, datasets, and models.

(e) **GPT-4o-mini (Achiam et al., 2023):** GPT-4o-mini is a compact multimodal reasoning model released by OpenAI in July 2024, delivering GPT-4–level performance while costing over 60% less than GPT-3.5 Turbo; it supports text and vision inputs, advanced function calling, and extended long-context understanding.

(f) **o4-mini (Achiam et al., 2023):** o4-mini is OpenAI's latest release of a reasoning-focused GPT variant that replaces o3-mini, offering both text and image processing, "whiteboard" chain-of-thought reasoning, seamless tool integration, and a high-accuracy paid-tier option—all accessible via ChatGPT and the Completions API for domain-critical decision-making tasks.

D Baseline Implementation

(a) **CDEs:** We follow ODEFormer's baseline implementation and hyperparameter tuning (d'Ascoli et al., 2023), using PySR with finite-difference approximations for skeleton search, and default greedy top-k generation for ODEFormer.

(b) **BNs:** We reimplement LogicGep's Boolean-network inference using the same Geppy genetic-programming framework (Gao et al., 2022), but omit the continuous-to-binary discretization and MLP-based constraint stages, since our training traces are already binary.

(c) **SCMs:** All SCM baselines are based on Tigramite (Runge et al., 2022), with the maximum time-lag set to 1 for fair comparison.

E Analysis on Table 2

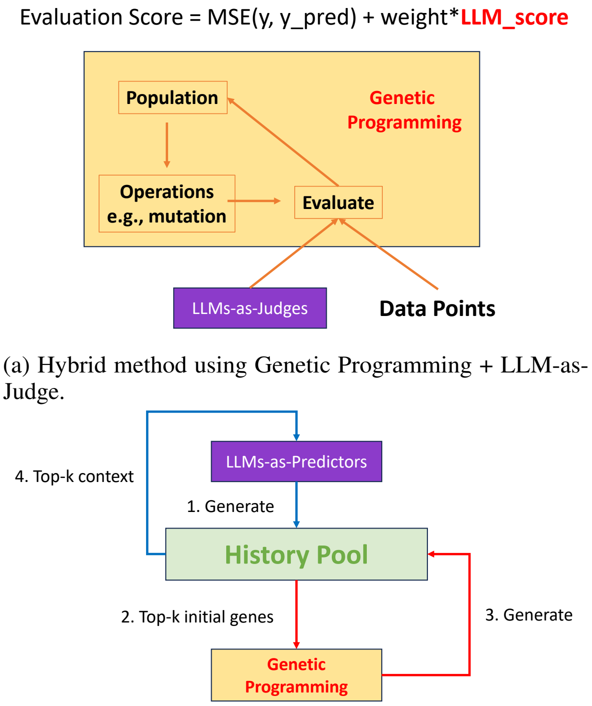
Prior studies (Shojaee et al., 2024; Merler et al., 2024) often use complexity as the sole standard for final selections of candidate predictions. However, we show that the generalization, represented by the performance during holdout evaluation (OOD), has a poor correlation with expression complexity. (a) As shown in Table 9, without introducing context, complexity may have a moderate correlation with $ACC_0.9$ when the dimension is small and not challenging. However, given samples with higher dimensions, the correlation became positive, meaning higher complexity can also give better generalization. (b) When context is introduced, besides the overall improved performance is observed as illustrated in **Obs. 4**, the correlation also turned positive from Dim=1 to Dim=3 and remained low for Dim=4. **Such an obvious change potentially suggests that context is a more effective criterion for candidate ranking and selection during both iterative refinement and final evaluation.**

Table 9: Correlations between complexity and OOD $ACC_{0.9}$ across four dimensions with and without context

Condition	Dim 1	Dim 2	Dim 3	Dim 4
w/o context	–0.672	–0.384	–0.425	0.584
w/ context	0.097	0.728	0.165	–0.174

F Hybrid Method

We adopt two frameworks for hybrid approach by separately let make GP and LLMs play different roles. (a) As shown in Figure 5a, in addition to the quantitative evaluation using MSE, an additional qualitative score produced by LLM is incorporated in the evolution loop; (b) As shown in Figure 5b, in addition to the original closed loop (blue line), we provide an extended path (red line) that utilize GP to expand the history pool. From a different perspective, the initial population produced by LLMs also improves the generation for GP by providing context-enhanced initial populations.



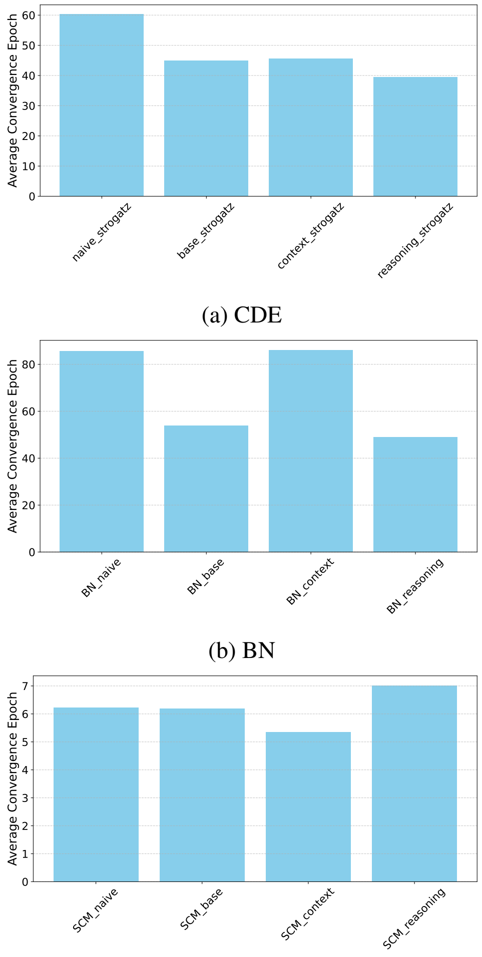
(a) Hybrid method using Genetic Programming + LLM-as-Judge.

(b) Hybrid method using Genetic Programming + LLM-as-Predictor, where GP helps expand the history pool with the current best expressions as initial population.

Figure 5: Hybrid method

G Convergence Rate

We further examine the convergence rate under various settings. As shown in Figure 6, introducing both context and reasoning leads to a faster convergence rate on the CDE and BN datasets. In contrast, the SCM dataset exhibits an overall fast convergence rate across all settings, with context having only a marginal improvement.



(a) CDE

(b) BN

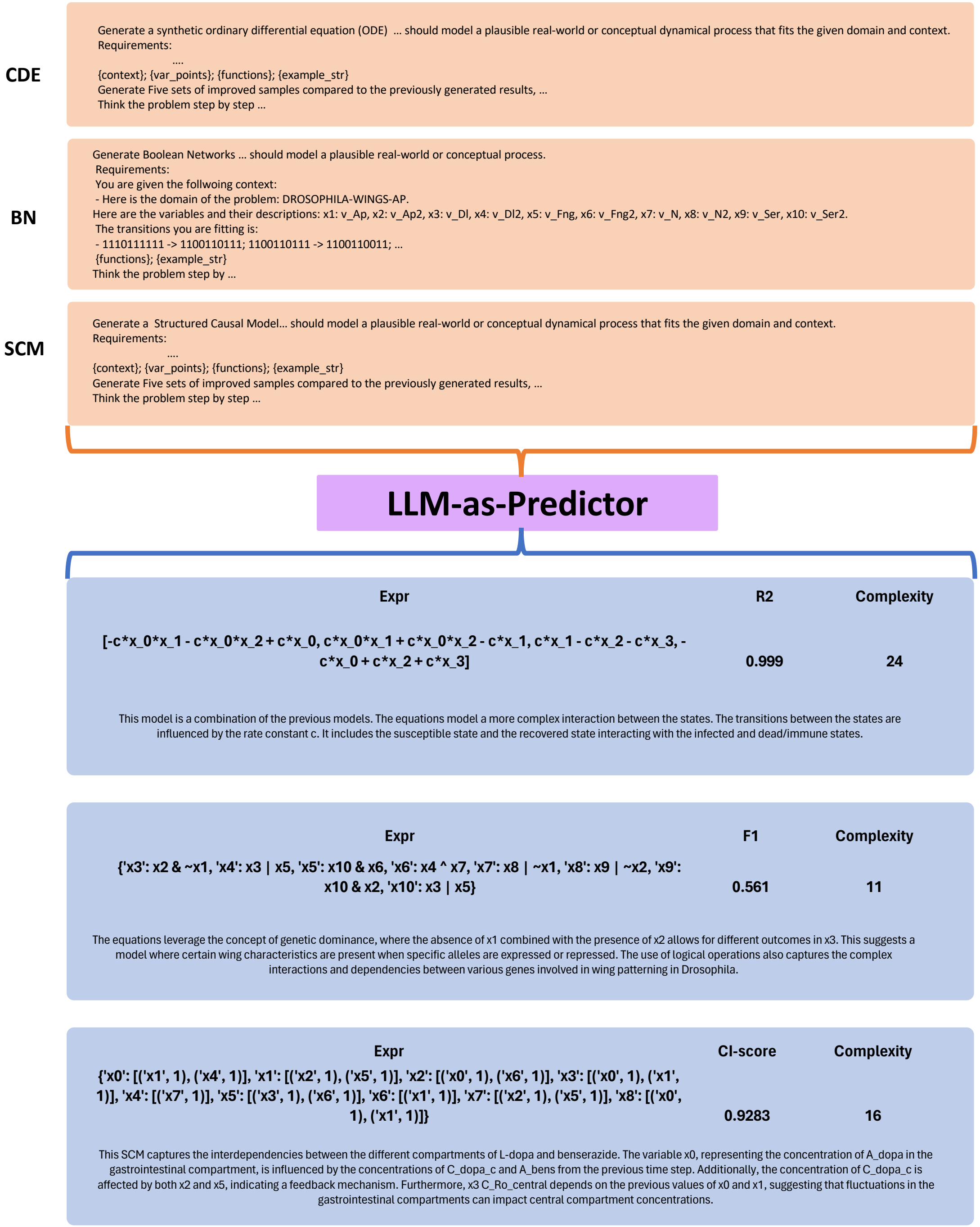
(c) SCM

Figure 6: Convergence Rate on CDE, BN, and SCM datasets.

H Example Outputs

H.1 Example predictions across three tasks

We present examples across three tasks in Table 8.



Generate a synthetic ordinary differential equation (ODE) ... should model a plausible real-world or conceptual dynamical process that fits the given domain and context.
Requirements:
....
{context}; {var_points}; {functions}; {example_str}
Generate Five sets of improved samples compared to the previously generated results, ...
Think the problem step by step ...

Generate Boolean Networks ... should model a plausible real-world or conceptual process.
Requirements:
You are given the follwoing context:
- Here is the domain of the problem: DROSOPHILA-WINGS-AP.
Here are the variables and their descriptions: x1: v_Ap, x2: v_Ap2, x3: v_Dl, x4: v_Dl2, x5: v_Fng, x6: v_Fng2, x7: v_N, x8: v_N2, x9: v_Ser, x10: v_Ser2.
The transitions you are fitting is:
- 1110111111 -> 1100110111; 1100110111 -> 1100110011; ...
{functions}; {example_str}
Think the problem step by ...

Generate a Structured Causal Model... should model a plausible real-world or conceptual dynamical process that fits the given domain and context.
Requirements:
....
{context}; {var_points}; {functions}; {example_str}
Generate Five sets of improved samples compared to the previously generated results, ...
Think the problem step by step ...

Expr	R2	Complexity
[-c*x_0*x_1 - c*x_0*x_2 + c*x_0, c*x_0*x_1 + c*x_0*x_2 - c*x_1, c*x_1 - c*x_2 - c*x_3, -c*x_0 + c*x_2 + c*x_3]	0.999	24

This model is a combination of the previous models. The equations model a more complex interaction between the states. The transitions between the states are influenced by the rate constant c. It includes the susceptible state and the recovered state interacting with the infected and dead/immune states.

Expr	F1	Complexity
{'x3': x2 & ~x1, 'x4': x3 | x5, 'x5': x10 & x6, 'x6': x4 ^ x7, 'x7': x8 | ~x1, 'x8': x9 | ~x2, 'x9': x10 & x2, 'x10': x3 | x5}	0.561	11

The equations leverage the concept of genetic dominance, where the absence of x1 combined with the presence of x2 allows for different outcomes in x3. This suggests a model where certain wing characteristics are present when specific alleles are expressed or repressed. The use of logical operations also captures the complex interactions and dependencies between various genes involved in wing patterning in Drosophila.

Expr	CI-score	Complexity
{'x0': [('x1', 1), ('x4', 1)], 'x1': [('x2', 1), ('x5', 1)], 'x2': [('x0', 1), ('x6', 1)], 'x3': [('x0', 1), ('x1', 1)], 'x4': [('x7', 1)], 'x5': [('x3', 1), ('x6', 1)], 'x6': [('x1', 1)], 'x7': [('x2', 1), ('x5', 1)], 'x8': [('x0', 1), ('x1', 1)]}	0.9283	16

This SCM captures the interdependencies between the different compartments of L-dopa and benserazide. The variable x0, representing the concentration of A_dopa in the gastrointestinal compartment, is influenced by the concentrations of C_dopa_c and A_bens from the previous time step. Additionally, the concentration of C_dopa_c is affected by both x2 and x5, indicating a feedback mechanism. Furthermore, x3 C_Ro_central depends on the previous values of x0 and x1, suggesting that fluctuations in the gastrointestinal compartments can impact central compartment concentrations.

Figure 7: Example input and output across three tasks using LLMs-as-Predictors. The output is processed through verification.

H.2 Candidates of CoT reasoning output on coupled-differential equations

We show the list of candidates found for fitting the time series generated from the SEIR model in Table 10.

Table 10: Example candidates of a 4-dimensional coupled differential equation during inference.

#	expr	R2	Complexity	reasoning (Qwen2.5-14B)	context_alignment	scientific_plausibility	conciseness&clarity	logical_coherence
0	{-c*x_0*x_1 - c*x_0*x_2 + c*x_0, c*x_0*x_1 + c*x_0*x_2 - c*x_1, c*x_1 - c*x_2 - c*x_3, -c*x_0 + c*x_2 + c*x_3}	0.99999	24	This model combines previous models. It introduces complex interaction between states. Transitions are governed by rate constant c. Susceptible and recovered states interact with infected/immune.	4	4	3	4
1	{c*x_0*x_1 - c*x_2 + c*x_3, c*x_1 + c*x_2 - c*x_3, -c*x_0 + c*x_2 + c*x_3, c*x_0 - c*x_1 + c*x_3}	0.99976	21	x_0 and x_1 interact to affect x_2. Other variable pairs interact linearly. This might reflect a chain of influence among factors—each influencing the next in a progression.	4	3	2	3
2	{c*x_0 + c*x_1 - c*x_2, c*x_1 + c*x_2 - c*x_3, -c*x_0 + c*x_2 + c*x_3, c*x_0 - c*x_1 + c*x_3}	0.99975	20	This system maintains cyclic balance among variables. Represents processes like ecological cycling where each state influences and is influenced in turn, sustaining continuous transitions.	2	2	3	2
3	{c*x_0*x_1 - c*x_0*x_2, c*x_0*x_2 - c*x_1, c*x_1 + c*x_2 - c*x_3, -c*x_0 + c*x_3}	0.99825	17	A variant epidemiological model. Introduces a slightly more intricate structure for transitions. Dynamics still guided by rate constant c. Greater attention to susceptible and infectious interplay.	4	4	3	4
4	{c*x_0*x_1 - c*x_2 + c*x_3, -c*x_0*x_2 + c*x_1 + c*x_3, c*x_0 - c*x_1*x_3 + c*x_2, -c*x_0*x_2 + c*x_1 + c*x_3}	0.99292	24	All variables appear in every equation. Introduces nonlinear terms, enabling feedback loops. May model biochemical systems with mutual influence and delayed or amplified effects.	4	3	2	3
5	{c*x_0*x_1 - c*x_2, c*x_1*x_2 - c*x_3, -c*x_0*x_1 + c*x_2, c*x_0 - c*x_1 + c*x_3}	0.96474	17	Models disease spread (SIR). x0: susceptible, x1: exposed, x2: infected, x3: recovered. Each equation captures standard transitions. E.g., x0*x1 for infections; x2 affects x3.	5	4	4	4
6	{c*x_0 - c*x_1*x_2, c*x_1*x_2 - c*x_3, -c*x_0 + c*x_2*x_3, -c*x_0 + c*x_1 + c*x_3}	0.90896	17	Similar to the SIR model with slight variations. Includes multiplicative terms between states. x0 and x1 drive infection; x2 and x3 influence recovery rates. Captures more nuanced behavior.	4	3	4	4
7	{-c*x_0*x_1 - c*x_0*x_2 + c*x_0, c*x_0*x_1 - c*x_1, c*x_0*x_2 - c*x_2, c*x_1 + c*x_2 - c*x_3}	0.79325	20	Basic epidemiological structure (SIR). Variables: x0: susceptible, x1: infected, x2: recovered, x3: immune/dead. Rate constant c influences all transitions between compartments.	4	4	3	4
8	{-c*x_0*x_1 + c*x_0*x_2 - c*x_2, c*x_0*x_1 - c*x_1 - c*x_2, c*x_2 - c*x_3, -c*x_0*x_2 + c*x_1}	0.33899	20	Represents SEIR model. dx0/dt: loss via infection, dx1/dt: gain from infection, loss via exposure, dx2/dt: gain from exposure, loss via recovery, dx3/dt: gain from recovery, potential loss.	4	4	3	4

H.3 RLM output for one sample with marked reasoning steps

(**Step 1: Problem Setup**) Okay, let's see. I need to generate a system of coupled ODEs for the calcium_dynamics domain. The variables are x_0, x_1, x_2, x_3, each with their descriptions as given. The time series data is provided, which includes x_0 values and the other variables. (**Step 2: Initial Observations**) First, the variables x_0, x_1, x_2, x_3 are each in the range of their description. The domain is calcium dynamics, so the equations need to model some process related to calcium concentration. Looking at the time series for x_0, which is a negative value, and the other variables being positive. Since calcium concentrations are usually positive, maybe the model is about the flow of calcium through the cell membrane, or the impact of different ions on calcium levels. The time series for x_1 is all zeros except at some points. The x_1 values are all 0.3, then increase. Maybe this is a transient phase or a step change. For x_2 and x_3, they seem to be varying linearly. (**Step 3: First Model Attempt**) Looking at the equations, perhaps there are some nonlinear terms. The variables can be combined using trigonometric functions. Let me think about a possible model. If I consider a calcium channel model, the rates of change could involve the concentration of ions. For example, the rate of change of x_0 might depend on x_1, x_2, and x_3. Similarly, x_1 might relate to x_2 and x_3. (**Step 4: Linear Combination Trial**) But the problem is to create a model that fits the data. Let's think of a simple model. Suppose that $x_0 = c(x_1 + x_2 + x_3)$. Similarly, maybe $x_1 = c(x_2 - x_3)$. For x_2, maybe it's a linear function: $x_2 = c(x_0 + x_3)$. Similarly, $x_3 = c(x_3 - x_0)$. Putting it all together, the system would be:

```
c*x1 + c*x2 + c*x3 | c*x2 - c*x1 | c*x0 + c*x3 | c*x3 - c*x0
```

(**Step 5: Plugging in Sample Data**) In the first sample: $x_0 = -43.0$, $x_1 = 0.03$, etc.
$x_0 = c(0.03 + 0.1 + 0.434) = c(0.564) \Rightarrow c = \frac{-43.0}{0.564} \approx -76.5$
$x_1 = c(0.1 - 0.434) = c(-0.334) \Rightarrow c = \frac{0.03}{-0.334} \approx -0.09$
Inconsistent values suggest equation mismatch. (**Step 6: Revising the Model**) Try again with the same structure:
$c = \frac{x_0}{x_1+x_2+x_3} = \frac{-43.0}{0.564} \approx -76.5$ $c = \frac{0.03}{0.1-0.434} = \frac{0.03}{-0.334} \approx -0.09$
Still inconsistent. (**Step 7: Trying Differential Form**) Suppose:

```
dx0/dt = c*(x1 + x2 + x3)
dx1/dt = c*(x2 - x3)
dx2/dt = c*(x0 + x3)
dx3/dt = c*(x3 - x0)
```

(**Step 8: Numerical Check**) With $x_0 = -43.0, x_1 = 0.03, x_2 = 0.1, x_3 = 0.434$:

```
dx0/dt = c*0.564
dx1/dt = c*(-0.334)
dx2/dt = c*0.464
dx3/dt = c*43.434
```

Still inconsistent c values. (**Step 9: Considering Other Forms**) Maybe use other terms, e.g., absolute values or trigonometric forms.

Step 10: Trying Absolute Values

```
dx0/dt = |x1 + x2 + x3|
dx1/dt = |x2 - x3|
dx2/dt = |x1 - x3|
dx3/dt = |x3 - x0|
```

Still not matching data. (**Step 11: Simpler Model Revisited**) Try:

```
dx0/dt = c*x1 + c*x2 + c*x3
dx1/dt = c*x2 - c*x3
dx2/dt = c*x0 + c*x3
dx3/dt = c*x3 - c*x0
```

(**Step 12: Integrating into Sample**) With $c = 1$:

```
dx0/dt = 0.564
dx1/dt = -0.334
dx2/dt = -42.566
dx3/dt = 43.434
```

(**Step 13: Final Justification**) Suggests calcium dynamics via flow of ions. x_0 as calcium concentration, x_1, x_2, x_3 as inflow/outflow.

(**Step 14: Final Model Form and JSON**)

```
dx0/dt = c*x1 + c*x2 + c*x3
dx1/dt = c*x2 - c*x3
dx2/dt = c*x0 + c*x3
dx3/dt = c*x3 - c*x0
```

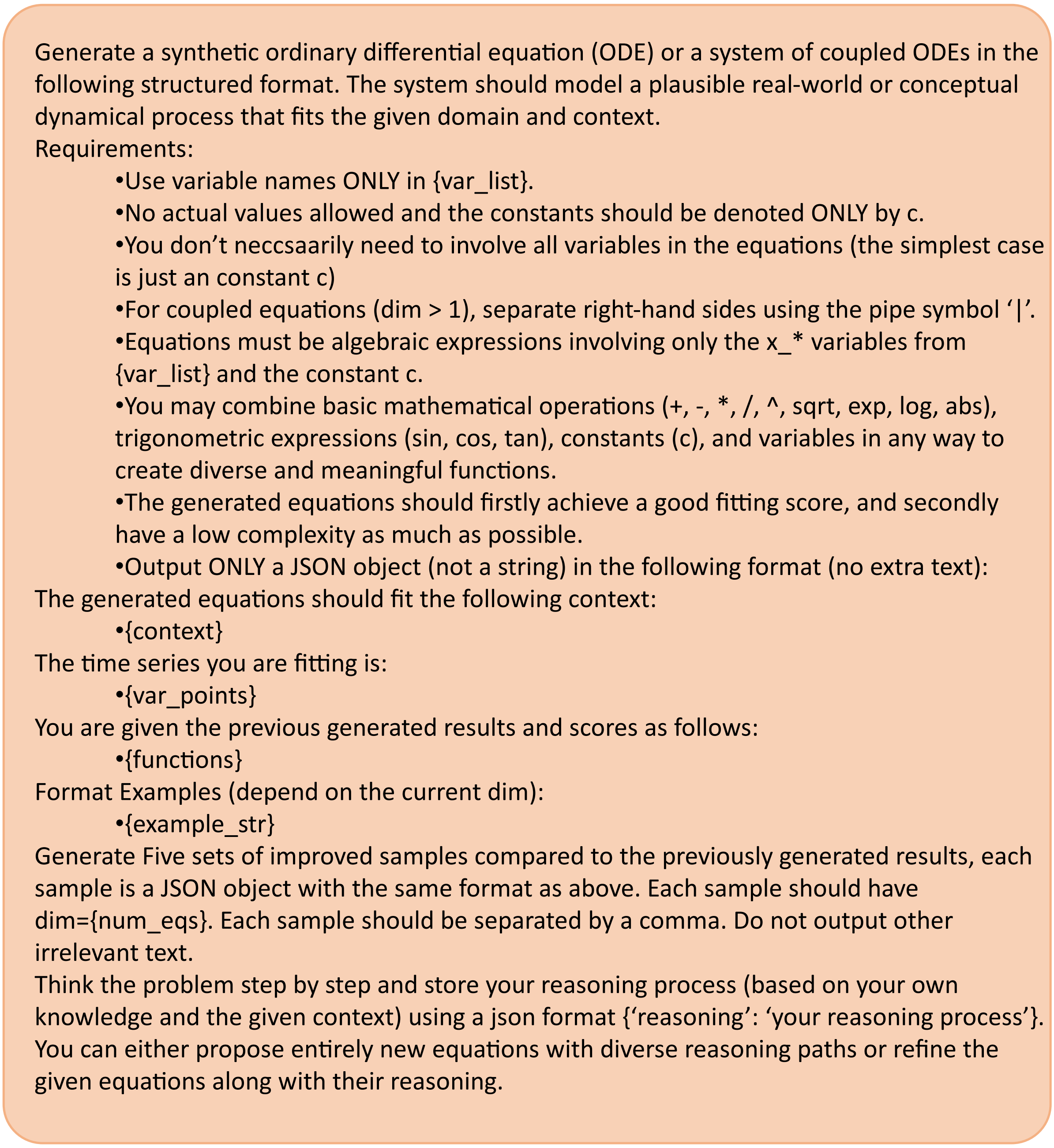
(**Step 15: Format Verification**) Follow pipe-separated format:

```
{"eq": "c*x_0*x_1 + c*x_2 - c*x_3 | c*x_1/x_0 + c*x_3 | c*x_2 - c*x_0*x_3 | c*x_3 + c*x_0 - c*x_1", "dim": 4}
```

(**Final JSON Output**)

```
{
  "eq": "c*x_0*x_1 + c*x_2 - c*x_3 | c*x_1/x_0 + c*x_3 | c*x_2 - c*x_0*x_3 | c*x_3 + c*x_0 - c*x_1",
  "dim": 4
}
```

I Prompts



Generate a synthetic ordinary differential equation (ODE) or a system of coupled ODEs in the following structured format. The system should model a plausible real-world or conceptual dynamical process that fits the given domain and context.
Requirements:

- Use variable names ONLY in {var_list}.
- No actual values allowed and the constants should be denoted ONLY by c.
- You don’t neccsaarily need to involve all variables in the equations (the simplest case is just an constant c)
- For coupled equations (dim > 1), separate right-hand sides using the pipe symbol ‘|’.
- Equations must be algebraic expressions involving only the x_* variables from {var_list} and the constant c.
- You may combine basic mathematical operations (+, -, *, /, ^, sqrt, exp, log, abs), trigonometric expressions (sin, cos, tan), constants (c), and variables in any way to create diverse and meaningful functions.
- The generated equations should firstly achieve a good fitting score, and secondly have a low complexity as much as possible.
- Output ONLY a JSON object (not a string) in the following format (no extra text):

The generated equations should fit the following context:

- {context}

The time series you are fitting is:

- {var_points}

You are given the previous generated results and scores as follows:

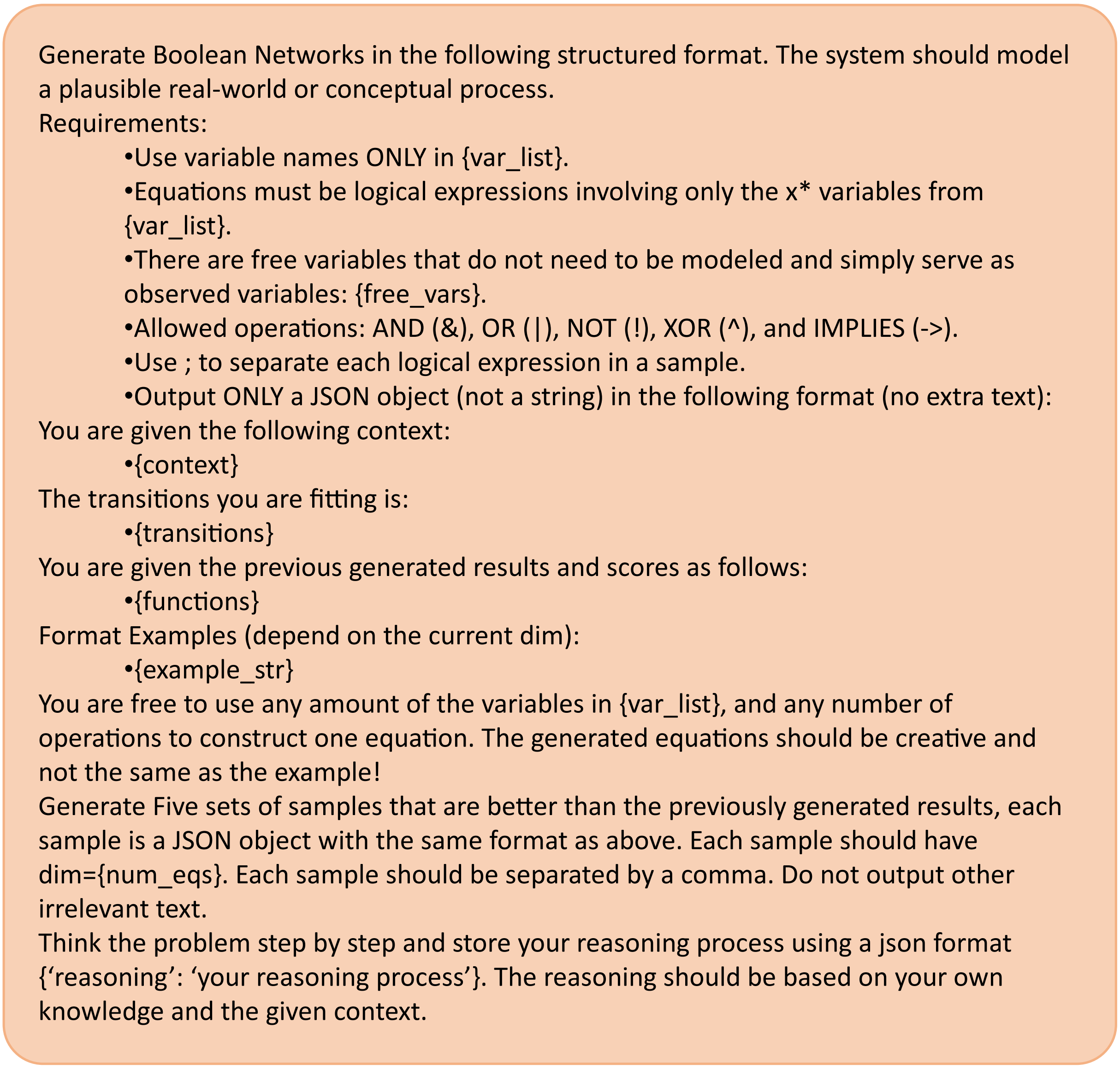
- {functions}

Format Examples (depend on the current dim):

- {example_str}

Generate Five sets of improved samples compared to the previously generated results, each sample is a JSON object with the same format as above. Each sample should have dim={num_eqs}. Each sample should be separated by a comma. Do not output other irrelevant text.
Think the problem step by step and store your reasoning process (based on your own knowledge and the given context) using a json format {‘reasoning’: ‘your reasoning process’}. You can either propose entirely new equations with diverse reasoning paths or refine the given equations along with their reasoning.

Figure 8: Prompt with CoT and Context for CDEs.



Generate Boolean Networks in the following structured format. The system should model a plausible real-world or conceptual process.

Requirements:

- Use variable names ONLY in {var_list}.
- Equations must be logical expressions involving only the x* variables from {var_list}.
- There are free variables that do not need to be modeled and simply serve as observed variables: {free_vars}.
- Allowed operations: AND (&), OR (|), NOT (!), XOR (^), and IMPLIES (->).
- Use ; to separate each logical expression in a sample.
- Output ONLY a JSON object (not a string) in the following format (no extra text):

You are given the following context:

- {context}

The transitions you are fitting is:

- {transitions}

You are given the previous generated results and scores as follows:

- {functions}

Format Examples (depend on the current dim):

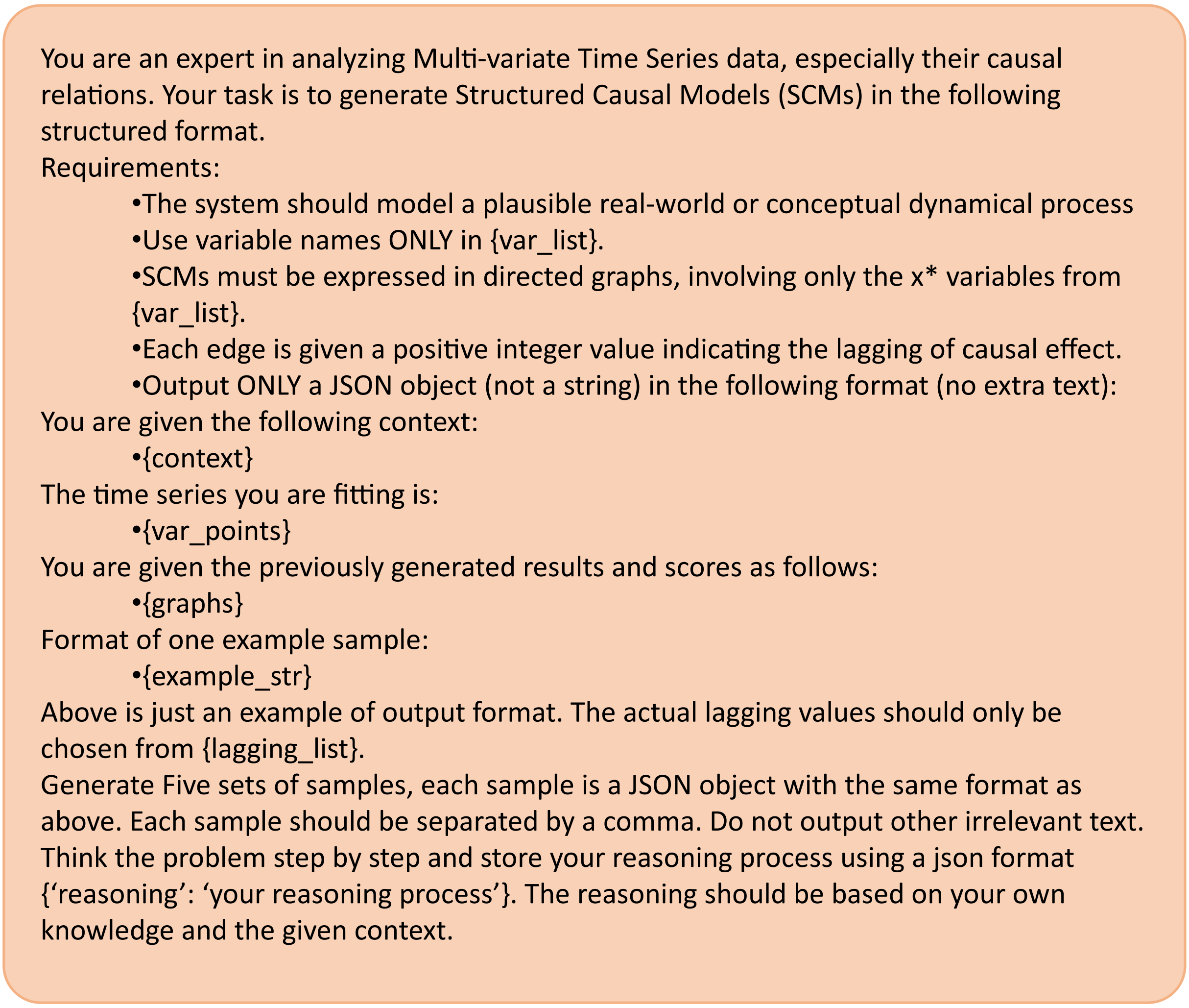
- {example_str}

You are free to use any amount of the variables in {var_list}, and any number of operations to construct one equation. The generated equations should be creative and not the same as the example!

Generate Five sets of samples that are better than the previously generated results, each sample is a JSON object with the same format as above. Each sample should have dim={num_eqs}. Each sample should be separated by a comma. Do not output other irrelevant text.

Think the problem step by step and store your reasoning process using a json format {‘reasoning’: ‘your reasoning process’}. The reasoning should be based on your own knowledge and the given context.

Figure 9: Prompt with CoT and Context for BNs.



You are an expert in analyzing Multi-variate Time Series data, especially their causal relations. Your task is to generate Structured Causal Models (SCMs) in the following structured format.

Requirements:

•The system should model a plausible real-world or conceptual dynamical process

•Use variable names ONLY in {var_list}.

•SCMs must be expressed in directed graphs, involving only the x* variables from {var_list}.

•Each edge is given a positive integer value indicating the lagging of causal effect.

•Output ONLY a JSON object (not a string) in the following format (no extra text):

You are given the following context:

•{context}

The time series you are fitting is:

•{var_points}

You are given the previously generated results and scores as follows:

•{graphs}

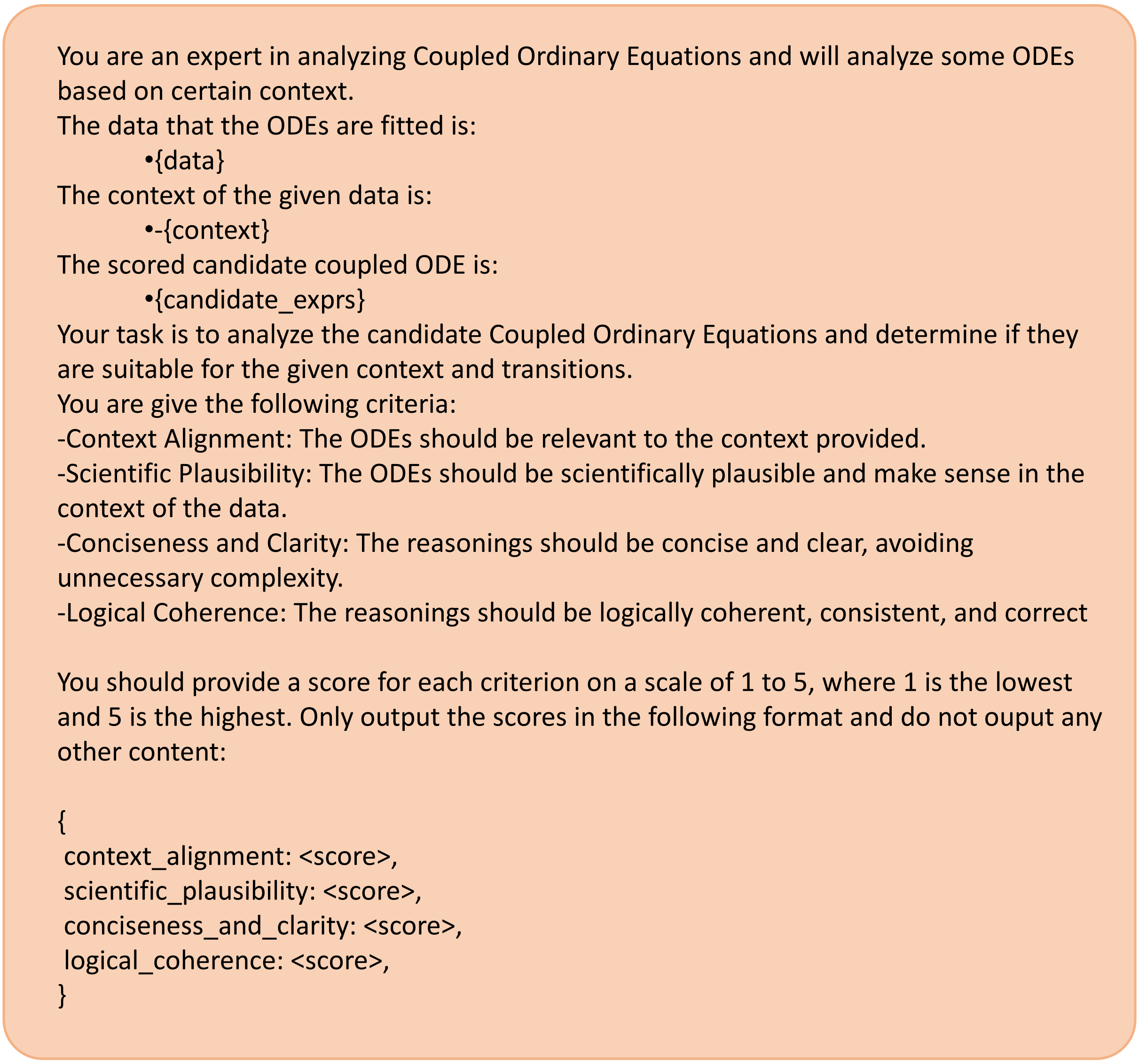
Format of one example sample:

•{example_str}

Above is just an example of output format. The actual lagging values should only be chosen from {lagging_list}.

Generate Five sets of samples, each sample is a JSON object with the same format as above. Each sample should be separated by a comma. Do not output other irrelevant text. Think the problem step by step and store your reasoning process using a json format {‘reasoning’: ‘your reasoning process’}. The reasoning should be based on your own knowledge and the given context.

Figure 10: Prompt with CoT and Context for SCMs.



You are an expert in analyzing Coupled Ordinary Equations and will analyze some ODEs based on certain context.
The data that the ODEs are fitted is:
•{data}
The context of the given data is:
•-{context}
The scored candidate coupled ODE is:
•{candidate_exprs}
Your task is to analyze the candidate Coupled Ordinary Equations and determine if they are suitable for the given context and transitions.
You are give the following criteria:
-Context Alignment: The ODEs should be relevant to the context provided.
-Scientific Plausibility: The ODEs should be scientifically plausible and make sense in the context of the data.
-Conciseness and Clarity: The reasonings should be concise and clear, avoiding unnecessary complexity.
-Logical Coherence: The reasonings should be logically coherent, consistent, and correct

You should provide a score for each criterion on a scale of 1 to 5, where 1 is the lowest and 5 is the highest. Only output the scores in the following format and do not ouput any other content:

```
{
 context_alignment: <score>,
 scientific_plausibility: <score>,
 conciseness_and_clarity: <score>,
 logical_coherence: <score>,
}
```

Figure 11: LLM-as-Judge prompt for CDEs.