

---

# DrSR: LLM based Scientific Equation Discovery with Dual Reasoning from Data and Experience

---

**Runxiang Wang<sup>1,2</sup> Boxiao Wang<sup>2,4</sup> Kai Li<sup>2,3\*</sup> Yifan Zhang<sup>1,2\*</sup> Jian Cheng<sup>1,2</sup>**

<sup>1</sup>School of Advanced Interdisciplinary Sciences, University of Chinese Academy of Sciences

<sup>2</sup>C<sup>2</sup>DL, Institute of Automation, Chinese Academy of Sciences

<sup>3</sup> School of Artificial Intelligence, University of Chinese Academy of Sciences

<sup>4</sup>School of Mathematical Sciences, University of Chinese Academy of Sciences

## Abstract

Symbolic regression is a fundamental tool for discovering interpretable mathematical expressions from data, with broad applications across scientific and engineering domains. Recently, large language models (LLMs) have demonstrated strong performance in this task, leveraging embedded scientific priors and reasoning capabilities to surpass traditional methods. However, existing LLM-based approaches, such as LLM-SR, often over-rely on internal priors, lacking explicit data understanding and systematic reflection during equation generation. To address these limitations, we propose DrSR (Dual Reasoning Symbolic Regression), a framework that combines data-driven insight with reflective learning to enhance both robustness and discovery capability. Specifically, DrSR guides LLMs to analyze structural relationships—e.g., monotonicity, nonlinearity, and correlation—within the data to generate structured descriptions. Simultaneously, it monitors equation performance and establishes a feedback loop to refine subsequent generations. By integrating data understanding and generation reflection in a closed loop, DrSR enables more efficient exploration of the symbolic expression space. Experiments across interdisciplinary datasets in physics, chemistry, biology, and materials science demonstrate that DrSR substantially improves the valid equation rate and consistently outperforms both classical and recent LLM-based methods in terms of accuracy, generalization, and search efficiency—underscoring its potential for scientific equation discovery.

## 1 Introduction

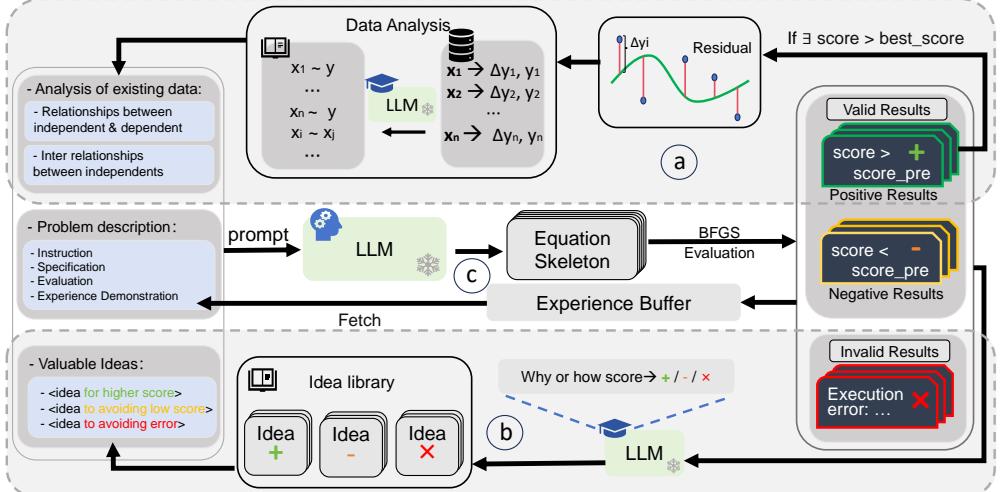
Symbolic regression (SR) [1] aims to recover interpretable mathematical expressions from observational data and plays a critical role in scientific modeling—ranging from uncovering physical laws [2, 3], to modeling chemical kinetics [4, 5], to understanding financial dynamics [6, 7]. The resulting equations support extrapolation, cross-domain transfer, and mechanistic understanding [8].

The core challenge in SR lies in navigating the vast, combinatorial space of mathematical expressions to identify models that both fit the data and remain interpretable. To address this, a range of algorithmic strategies have been proposed. Classical approaches based on genetic programming (GP) [8, 9] evolve expression trees via mutation and crossover. Reinforcement learning (RL) methods [10] treat symbolic equation discovery as a sequential decision-making problem, using policy gradients to explore the space of symbolic programs. Transformer-based neural models [11, 12] have been employed to directly generate symbolic expressions by modeling equations as sequences.

More recently, LLMs have further advanced this area by generating candidate equations from textual prompts. For example, LLM-SR [13] proposes new expressions conditioned on task descriptions and

---

\*Corresponding authors.



**Figure 1: Overview of the DrSR framework.** (a) *Data-aware Insight*: the LLM analyzes variable relationships to derive structured, data-informed insights; (b) *Inductive Idea Extraction*: based on equation fitness, the LLM summarizes successful and failed patterns into an evolving idea library; (c) *Equation Generation and Selection*: guided by (a) and (b), the LLM generates new equation skeletons, optimizes parameters, and caches the best-performing results to inform future generations.

prior candidates, combining LLM generation with numeric optimization. However, such methods overly rely on the scientific priors embedded within LLMs while neglecting structural insights into variable relationships—effectively ignoring direct observation and analysis of the raw data. This limits the accuracy and generalizability of the generated equations. Moreover, the lack of systematic reflection on generation history often leads to the repeated production of invalid expressions (e.g., syntax errors, numerical overflows), reducing both the efficiency and stability of SR. To address these issues, a more effective SR framework is needed—one that not only incorporates scientific priors flexibly, but also analyzes data structures and learns from past generation behavior. Such an approach would better align with the scientist’s cognitive process: inducing patterns from observations, reflecting through trial and error, and efficiently navigating the vast hypothesis space to uncover interpretable and predictive models.

To this end, we propose DrSR (Dual Reasoning Symbolic Regression) (Fig 1), a framework that augments LLM-based generation with two synergistic reasoning mechanisms: *data-aware insight*, which derives structural patterns from raw data, and *inductive idea extraction*, which reflects on generation outcomes to distill reusable strategies. Specifically, building upon the generation-evaluation loop of LLM-SR, a data-aware module prompts the LLM to extract structural patterns from a sampled subset of raw data, providing task-specific priors on variable relationships. As better-fitting equations are discovered, these insights are incrementally updated (Fig 1(a)). Secondly, DrSR introduces an experience-based module that categorizes newly generated equations based on evaluation outcomes and prompts the model to reflect on success or failure. The resulting strategies—summarized as structured “ideas”—are stored in a dynamic idea library and reused to improve future equation generation (Fig 1(b)). By combining data-driven interpretation with behavior-level feedback, DrSR enables continual refinement of both symbolic priors and generative heuristics, resulting in more efficient and reliable symbolic equation discovery.

We evaluate DrSR on six benchmarks spanning physics, chemistry, biology, and materials science, using Mixtral-8x7B-Instruct-v0.1 [14] and LLaMA3.1-8B-Instruct [15] as backbones. Compared to traditional SR methods and LLM-based baselines, DrSR achieves state-of-the-art performance across multiple metrics, including accuracy, generalization, valid expression rate, and convergence speed.

## 2 Preliminaries

In SR, the objective is to discover a compact symbolic expression  $f(\mathbf{x})$  that approximates an unknown mapping  $f^{real} : \mathbb{R}^d \rightarrow \mathbb{R}$  based on a dataset of input-output pairs  $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ . SR methods

aim to recover underlying mathematical relationships such that  $f(\mathbf{x}_i) \approx y_i$  for all  $i$ . The discovered expression is expected to not only fit the observed data accurately but also generalize well to unseen inputs while maintaining interpretability. Model performance is typically evaluated via the negative mean squared error:  $\text{Score}(f, D) = -\frac{1}{n} \sum_{i=1}^n (f(\mathbf{x}_i) - y_i)^2$ .

Our work builds upon LLM-SR [13], a recent framework that utilizes LLM to generate interpretable equations from data. To contextualize our proposed DrSR, we briefly review the core workflow of LLM-SR. LLM-SR discovers equations by combining the scientific knowledge embedded in LLMs with their powerful code generation capabilities, prompting the LLM to generate equation skeletons  $\mathcal{F}$  with learnable parameters. The overall procedure consists of three key components:

**Hypothesis Generation.** The LLM receives a structured prompt comprising: a task description  $\mathcal{T}$ , variable constraints, evaluation criteria, and reference equations. Based on this, it generates equation skeletons aligned with the physical meaning of variables.

**Hypothesis Evaluation.** Each skeleton is fitted to data by optimizing its parameters. The resulting function is scored via the fitness metric  $\text{Score}(f, D)$ .

**Experience Management.** A memory buffer stores high-scoring equations, which are reused as demonstrations in future prompts to improve generation quality.

While LLM-SR effectively leverages prior knowledge embedded in LLMs, it suffers from two critical drawbacks: First, its understanding of data remains shallow: although LLM-SR can generate equation skeletons via language prompts and refine them through parameter optimization, it lacks direct engagement with the data itself. As a result, it fails to capture complex inter-variable relationships that are crucial for SR, limiting the reasoning capacity of LLMs. Second, it lacks a reflective mechanism to evaluate and learn from its own behavior. Without such self-correction, LLM-SR frequently produces invalid expressions—such as syntactically incorrect forms, numerical instabilities, or inconsistent variable usage—ultimately degrading performance and robustness.

### 3 Method

To overcome these limitations, we propose DrSR (Dual Reasoning Symbolic Regression), a novel framework that enhances SR by equipping LLM with two synergistic reasoning pathways: *Data-aware Insight* and *Inductive Idea Extraction*. While retaining the generative strength of LLM-SR in leveraging scientific priors, DrSR introduces a closed-loop reasoning cycle that enables the model to both analyze raw data and reflect on its generation history. DrSR transforms SR into a dynamic cognitive process: observing to understand and reflecting to improve.

To implement this, DrSR instantiates three role-specific, LLM-based modules:  $\pi_{\text{data}}$  for data-aware insight extraction,  $\pi_{\text{idea}}$  for modeling experience summarization, and  $\pi_{\text{main}}$  for equation generation. All modules share the same LLM backbone.

#### 3.1 Data-aware Insight

To enhance the structural awareness of SR, we introduce a dedicated *Data-Analysis-LLM*  $\pi_{\text{data}}$ , which analyzes input-output pairs and residuals to generate structured, iteratively updated insights  $\mathcal{D}$ . These insights encode variable relationships and serve as evolving structural priors for the main model  $\pi_{\text{main}}$  to guide equation generation.

##### 3.1.1 Initial Data-aware Insight

To control input length and preserve key behavioral patterns, we uniformly sample 100 input-output pairs from the original dataset  $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$  to form  $D_0^{\text{resample}}$ . Feeding this into  $\pi_{\text{data}}$  yields the initial insight  $\mathcal{D}_0$ , capturing basic patterns such as monotonicity, nonlinearity, and variable correlation.

##### 3.1.2 Iterative Residual-based Refinement

At each iteration  $t$ , if a candidate  $f^*$  improves upon the current best score  $s^*$ ,  $\pi_{\text{data}}$  produces an updated insight  $\mathcal{D}_{\text{new}}$  to refine the future generations of  $\pi_{\text{main}}$ .

**Prompt Construction.** Residuals provide informative signals about parts of the data that are poorly captured by the current model, often revealing latent variable interactions or unmodeled structures. To help  $\pi_{\text{data}}$  focus on these regions, we compute residuals as  $rest_{t,i} = y_i - f^*(\mathbf{x}_i)$  and construct an augmented dataset  $D_t = \{(\mathbf{x}_i, y_i, rest_{t,i})\}_{i=1}^n$ . We then uniformly resample 100 examples from  $D_t$  to form  $D_t^{\text{resample}}$ , which is used to generate refined structural insights in the next iteration.

**Insight Generation.** Using  $D_t^{\text{resample}}$ , the candidate  $f^*$ , and prior insight  $\mathcal{D}_{\text{pre}}$ ,  $\pi_{\text{data}}$  generates a refined interpretation. This includes local monotonicity changes, nonlinear interactions (e.g., products, ratios), and more specific variable correlations, forming a stronger prior for guiding generation.

As iterations proceed, insights evolve from coarse global patterns to finer local and higher-order structures, guiding  $\pi_{\text{main}}$ 's generation process toward more precise, interpretable expressions.

### 3.2 Inductive Idea Extraction

To improve search efficiency and robustness, we introduce an *Idea-Extraction-LLM*  $\pi_{\text{idea}}$ , which mimics scientific reflection by extracting actionable ideas from model feedback. At each iteration, it analyzes the generated equations and summarizes both successes and failures into structured heuristics. These ideas encode generation strategies, common pitfalls, or error-avoidance techniques, enabling the model to iteratively refine its behavior and improve symbolic discovery.

#### 3.2.1 Extraction Process

After  $\pi_{\text{main}}$  generates and optimizes equations, each evaluated candidate  $f$  is categorized as: **1) Positive.** Outperforms the reference, suggesting effective modeling. **2) Negative.** Performs worse than the reference, indicating potential flaws. **3) Invalid.** Cannot be evaluated due to syntax errors, numerical faults, or variable mismatches.

For each type,  $\pi_{\text{idea}}$  is prompted with tailored templates (see Appendix C) to elicit structural heuristics: e.g., strengths of winning patterns, causes of performance drops, or failure modes and remedies. The extracted ideas are categorized and stored for downstream use.

---

#### Algorithm 1 DrSR

---

**Input:** LLM  $\pi_{\text{data}}$ , LLM  $\pi_{\text{idea}}$ , LLM  $\pi_{\text{main}}$ , dataset  $D$ ,  $T$  iterations,  $k$  in-context examples,  $b$  samples per iteration, proportion of recent ideas  $\lambda$

```

 $P_0, \mathcal{D}, \mathcal{L}, \mathcal{P} \leftarrow \text{INITALL}()$ 
 $f^*, s^* \leftarrow \text{null}, -\infty$ 
for  $t = 1$  to  $T$  do
     $\mathcal{F}_t \leftarrow \text{SAMPLELLM}(\pi_{\text{main}}, \mathcal{P}, k, b)$ 
    for  $f \in \mathcal{F}_t$  do
         $s \leftarrow \text{Score}(f, D)$ 
         $\mathcal{L} \leftarrow \text{IDEASUMMARIZATION}(f, \mathcal{P},$ 
         $\pi_{\text{idea}}, \mathcal{D}, s^*, \mathcal{L})$ 
        if  $s > s^*$  then
             $\mathcal{D} \leftarrow \text{DATAANALYSIS}(f, \mathcal{D},$ 
             $D, \pi_{\text{data}})$ 
             $f^*, s^* \leftarrow f, s$ 
             $P_t \leftarrow P_{t-1} \cup \{(f, s)\}$ 
        end if
    end for
    # Sample from latest  $\lambda$  proportion
     $\mathcal{L}_{\text{sampled}} \leftarrow \text{SAMPLERECENTIDEAS}(\mathcal{L}, \lambda)$ 
     $\mathcal{P} \leftarrow \text{UPDATEPROMPT}(\mathcal{L}_{\text{sampled}}, \mathcal{D})$ 
end for
Output:  $f^*, s^*$ 

```

---



---

#### Algorithm 2 IDEA-SUMMARIZATION

---

**Input:** Equation  $f$ , prompt history  $\mathcal{P}$ , LLM  $\pi_{\text{idea}}$ , dataset  $D$ , best score  $s^*$ , Library  $\mathcal{L}$

```

result  $\leftarrow \text{EVALUATEEQUATION}(f, D)$ 
if result contains error then
    category  $\leftarrow \text{INVALID}$ 
else if  $\text{Score}(f, D) > s^*$  then
    category  $\leftarrow \text{POSITIVE}$ 
else
    category  $\leftarrow \text{NEGATIVE}$ 
end if
 $\mathcal{I} \leftarrow \text{IDEAEXTRACTION}(\mathcal{P}, f, \text{result}, \text{category}, \pi_{\text{idea}})$ 
 $\mathcal{L} \leftarrow \text{IDEALIBRARYUPDATE}(\mathcal{L}, \mathcal{I}, \text{category})$ 
Output:  $\mathcal{L}$ 

```

---



---

#### Algorithm 3 DATA-ANALYSIS

---

**Input:** Equation  $f$ , Dataset  $D$ , Previous insight  $\mathcal{D}_{\text{pre}}$ , LLM  $\pi_{\text{data}}$

```

 $\hat{y}_i = f(x_i; \theta^*)$ ,  $rest_{t,i} = y_i - \hat{y}_i$ 
 $D_t = \{(\mathbf{x}_i, y_i, rest_{t,i})\}_{i=1}^n$ 
# Resample:  $(x'_i, y'_i, rest'_{t,i}) \sim_{\text{i.i.d.}} \text{Unif}(D_t)$ 
 $D_t^{\text{resample}} = \{(x'_i, y'_i, rest'_{t,i})\}_{i=1}^{n'}$ 
 $\mathcal{D}_{\text{new}} \leftarrow \pi_{\text{data}}(D_t^{\text{resample}}, f, \mathcal{D}_{\text{pre}})$ 
Output:  $\mathcal{D}_{\text{new}}$ 

```

---

#### 3.2.2 Idea Library

Extracted ideas are organized in the Idea Library  $\mathcal{L}$  as follows:

**Positive Ideas:** Capture patterns from successful equations, such as the inclusion of high-order or domain-specific terms (e.g., sinusoidal or multiplicative components).

**Negative Ideas:** Highlight structural flaws—e.g., overcomplexity or mismatches with data characteristics—helping to steer the model away from ineffective regions.

**Error Avoidance Ideas:** Focus on eliminating invalid outputs by encoding rules that ensure syntactic and semantic correctness.

All ideas are stored in JSON format with metadata including context and fitness score. In each iteration, recently extracted ideas are sampled and injected into prompts of  $\pi_{\text{main}}$ , supporting continual adaptation and improved generation quality.

### 3.3 Overall Workflow of DrSR

DrSR integrates two complementary reasoning pathways—*data-driven insight* and *experience-based reflection*—to enable LLMs to perform SR with self-correcting and structurally informed capabilities. In each iteration, the model not only generates and evaluates candidate equations but also refines its understanding of the data and summarizes patterns from prior attempts. This yields a closed-loop optimization process that improves both equation quality and search efficiency over time. The pseudocode of DrSR is detailed in Algorithm 1.

### 3.4 A Bayesian Perspective on DrSR

The superior performance of DrSR arises not merely from architectural enhancements, but from a principled rethinking of how LLMs conduct SR. We reinterpret its generative process through the lens of Bayesian modeling to highlight its strengths over prior approaches such as LLM-SR.

In SR, the goal is to identify an expression  $f$  from dataset  $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$  that is both predictive and interpretable. Following the Bayesian formulation [16], LLM-SR can be viewed as maximizing:

$$\arg \max_f p_{\text{LLM}}(f | D) = \arg \max_f p_{\text{LLM}}(D | f) \cdot p_{\text{LLM}}(f), \quad (1)$$

where  $p_{\text{LLM}}(f)$  captures the prior belief of LLMs about the plausibility of expression  $f$ , reflecting how likely the LLM is to generate  $f$  based on its learned distribution over symbolic forms, and  $p_{\text{LLM}}(D | f)$  measures the explanatory adequacy of the expression  $f$  with respect to the dataset  $D$ . However, this approach relies heavily on language-induced priors and lacks two key elements: structural understanding of the data and reflective learning from prior generations.

DrSR addresses these limitations by introducing two adaptive cognitive variables: the data insight  $\mathcal{D}$ , summarizing variable relationships, and the idea library  $\mathcal{I}$ , capturing learned generation strategies. With these, the objective becomes:

$$\arg \max_{f, \mathcal{D}, \mathcal{I}} p_{\text{LLM}}(f, \mathcal{D}, \mathcal{I} | D) = \arg \max_{f, \mathcal{D}, \mathcal{I}} p_{\text{LLM}}(D | f) \cdot p_{\text{LLM}}(f | \mathcal{D}, \mathcal{I}) \cdot p_{\text{LLM}}(\mathcal{D}, \mathcal{I}), \quad (2)$$

where  $p_{\text{LLM}}(f | \mathcal{D}, \mathcal{I})$  serves as a cognitively enriched prior that guides equation generation, and  $p_{\text{LLM}}(\mathcal{D}, \mathcal{I})$  models the evolution of structural knowledge over time.

This formulation does not change the task objective but enhances the generative prior from a static, language-based prior  $p_{\text{LLM}}(f)$  to a dynamic, context-aware form conditioned on evolving insight and experience. As  $\mathcal{D}$  and  $\mathcal{I}$  are iteratively refined, the structural preferences of the model become sharper, enabling more efficient and stable exploration of the symbolic expression space.

## 4 Experiments

### 4.1 Benchmarks and Datasets

We evaluate DrSR on six representative SR datasets spanning physics, biology, chemistry, and materials science. Four datasets are taken from LLM-SR [13], and two from the LLM-SRBench suite [17], including LSR-Transform and LSR-Synth. Full details are available in Appendix B.

**Nonlinear Oscillators.** This dataset simulates two nonlinear damped oscillator systems, each governed by second-order differential equations involving displacement, velocity, and external

forcing. Compared to classical oscillators, they incorporate cubic, multiplicative, and non-polynomial terms, yielding complex variable dependencies and heightened modeling difficulty.

**Bacterial Growth.** This dataset models the growth rate of *E. coli* under varying conditions, incorporating population density, substrate concentration, temperature, and pH. The ground-truth equation involves nonlinear effects of temperature and pH, reflecting complex interactions among variables. The task is to recover expressions that capture the dynamic behavior of bacterial proliferation [18, 19].

**Stress–Strain Behavior.** This dataset captures the mechanical response of 6061-T651 aluminum alloy under varying strain and temperature, based on tensile tests conducted across six temperatures (20°C to 300°C). It serves as a realistic benchmark for modeling stress as a function of strain and thermal conditions, and is widely applicable in structural engineering and materials science.

**LSR-Transform.** This dataset, from LLM-SRBench, tests the ability of LLMs to recover structurally altered expressions. The target functions originate from Feynman equations [20], transformed via variable substitutions and symbolic rewrites to increase complexity. Due to computational constraints, we select two representative subsets—I.37.4\_0\_1 and III.4.33\_3\_0—for experimentation.

**LSR-Synth.** This synthetic benchmark from LLM-SRBench [17] includes 129 SR tasks constructed by mixing canonical scientific terms with synthetic expressions. The dataset spans four domains—chemistry, physics, biology, and materials science—and aims to test LLMs’ ability to generalize across hybrid symbolic forms. Due to domain overlap with [13] and compute limits, we focus on the chemistry subset CRKO.

## 4.2 Baselines

We compare DrSR with six representative SR methods, spanning classical, deep learning, and LLM-based paradigms: gplearn, PySR [9], DSR [10], uDSR [21], LaSR [22], and LLM-SR [13].

gplearn uses GP to represent symbolic expressions as trees and explores the hypothesis space via mutation and crossover. PySR extends GP with parallelization and heuristics like simulated annealing and adaptive parsimony to improve efficiency. DSR employs RL to train neural networks that generate symbolic formulas. uDSR enhances DSR by combining neural-guided GP with large-scale pretraining. LaSR integrates LLMs with GP, using zero-shot prompts to guide the evolution of abstract equation representations. LLM-SR treats equations as programs, generating equation skeletons with LLMs and refining them via evolutionary search. These baselines represent diverse SR approaches and serve as strong comparators to evaluate the effectiveness of DrSR.

## 4.3 Evaluation Metrics

We evaluate SR models using two complementary metrics: Accuracy under error tolerance ( $\text{ACC}_\tau$ ) and Normalized Mean Squared Error (NMSE), which respectively measure tolerance-aware generalization and numeric precision. Specifically  $\text{ACC}_\tau$  is defined as the proportion of test points with error below a threshold  $\tau$ :  $\text{ACC}_\tau = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \mathbf{1}\left(\left|\frac{f(x_i) - y_i}{y_i}\right| \leq \tau\right)$ , where  $\mathbf{1}(\cdot)$  is the indicator function.

NMSE quantifies prediction error normalized by target variance:  $\text{NMSE} = \frac{\sum_{i=1}^{N_{\text{test}}} (f(x_i) - y_i)^2}{\sum_{i=1}^{N_{\text{test}}} (y_i - \bar{y})^2}$ , where  $\bar{y}$  is the mean of the true values. NMSE places more weight on large deviations, making it suitable for evaluating numeric precision.  $\text{ACC}_\tau$  captures “good enough” performance under bounded error, aligning with real-world tolerance requirements. Together, these metrics provide a holistic view of model performance, balancing exactness and practicality.

## 4.4 DrSR Configuration

For fair comparison, we use the same backbone LLMs (Mixtral-8x7B-Instruct-v0.1 and LLAMA3.1-8B-Instruct) across all LLM-based methods. In each iteration, 4 candidate equation skeletons are sampled, and their parameters are optimized via BFGS. DrSR is evaluated with a maximum of 1000 iterations per dataset, while baselines like LLM-SR, LaSR, and classical methods are allowed 2000 iterations or more to ensure convergence. Further implementation details, including prompts and sampling settings, are provided in Appendices A and C.

Table 1: Overall performance of DrSR and baseline methods on six symbolic regression benchmarks.

Model	Oscillation 1		Oscillation 2		E. coli growth		Stress-Strain		LSR-Transform-2Avg		LSR-Synth-CRK0	
	Acc <sub>0.001</sub> (%)↑	NMSE↓	Acc <sub>0.001</sub> (%)↑	NMSE↓	Acc <sub>0.1</sub> (%)↑	NMSE↓	Acc <sub>0.1</sub> (%)↑	NMSE↓	Acc <sub>0.1</sub> (%)↑	NMSE↓	Acc <sub>0.1</sub> (%)↑	NMSE↓
GPlern	0.11	0.0972	0.05	0.2000	0.76	1.002	28.43	0.3496	86.67	0.0070	0.80	1.0373
PySR	3.80	0.0003	7.02	0.0002	2.80	0.4068	70.60	0.0347	50.57	0.0034	<b>100</b>	8.49e-7
DSR	0.42	0.01916	0.24	0.1793	0.64	0.8294	59.85	0.3435	70.94	0.0150	2.00	1.32e-5
uDSR	1.78	0.0002	0.36	0.0856	1.12	0.5059	59.15	0.0639	82.62	0.0012	98.60	1.44e-7
LLM-SR (Mixtral)	5.9	0.0001	7.62	4.59e-5	2.08	0.2282	68.10	0.0530	90.79	0.0036	77.60	1.09e-6
LLM-SR (Llama3.1)	12.67	2.55e-5	8.2	4.70e-5	1.36	0.5815	76.21	0.0333	93.73	0.0008	87.00	2.52e-6
LaSR (Mixtral)	1.99	0.3332	1.64	0.0058	2.32	0.2955	20.52	1.1923	91.94	0.0029	94.20	2.49e-6
LaSR (Llama-3.1)	2.79	0.7485	1.09	0.0310	3.44	0.1349	71.84	0.0320	92.30	0.0031	95.40	6.52e-8
DrSR (Mixtral)	<b>83.92</b>	<b>3.14e-7</b>	<b>99.94</b>	<b>1.80e-12</b>	<b>5.12</b>	<b>0.0195</b>	<b>88.28</b>	<b>0.0156</b>	<b>92.51</b>	<b>0.0055</b>	<b>95.20</b>	<b>8.87e-8</b>
DrSR (Llama-3.1)	77.98	5.40e-7	99.33	1.23e-9	3.64	0.0797	72.95	0.0230	<b>96.32</b>	<b>0.0006</b>	98.00	<b>1.75e-8</b>

## 5 Findings

### 5.1 Overall Performance Comparison

As shown in Table 1, DrSR consistently achieves significantly lower NMSE and higher Acc<sub>τ</sub> across most datasets, outperforming both classical and LLM-based baselines. For example, on Oscillator 2, DrSR (Mixtral) achieves an NMSE of  $1.80 \times 10^{-12}$ , far surpassing LLM-SR (Mixtral)'s  $4.59 \times 10^{-5}$ . In terms of accuracy, DrSR exceeds 90% on several benchmarks with both Mixtral and LLaMA backbones—highlighting strong alignment with ground truth under strict error thresholds.

Traditional methods such as PySR show reasonable performance on some tasks, such as LSR-Synth-CRK0, but struggle on others. Meanwhile, LLM-SR and LaSR, though capable of generating syntactically valid expressions, often suffer from instability and limited data sensitivity—failing to leverage feedback for continuous improvement.

These results underscore the core advantage of DrSR: its dual reasoning architecture. The *Data-aware Insight* module provides structural priors from inter-variable analysis, while the *Idea Extraction* module captures and reuses generation heuristics. Together, they enable more guided, interpretable, and efficient exploration of the expression space.

### 5.2 Generalization Evaluation

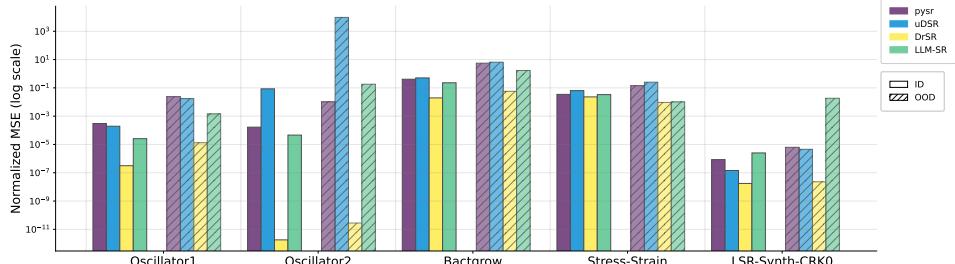


Figure 2: **Generalization across scientific domains under ID and OOD settings.** DrSR achieves the lowest NMSE across all tasks and settings, indicating strong generalization performance.

We further assess generalization by evaluating performance under both in-distribution (ID) and out-of-distribution (OOD) settings. Figure 2 presents the NMSE of PySR, uDSR, LLM-SR, and DrSR under both conditions. DrSR consistently achieves the best performance in both ID and OOD cases, confirming its robustness. Notably, on Oscillator 2, DrSR reduces OOD NMSE to  $2.80 \times 10^{-11}$ , whereas uDSR degrades to  $9.59 \times 10^3$ , indicating severe overfitting.

DrSR also exhibits minimal performance drop between ID and OOD, suggesting it learns latent rules behind data rather than overfitting to distribution-specific patterns. This generalization is critical for real-world scientific modeling, where extrapolation beyond observed data is often required. Taken together, these findings position DrSR as a robust and accurate SR framework, capable of producing generalizable, interpretable models across a wide range of scientific domains.

### 5.3 Training Efficiency Comparison

We evaluate the convergence efficiency of DrSR relative to LLM-SR, uDSR, and PySR on six benchmark tasks. Figure 3 shows the NMSE over training iterations. DrSR converges significantly faster than all baselines and achieves lower final errors. In most cases, it matches or surpasses the 2000-iteration performance of others within just 1000 steps. This efficiency stems from DrSR’s dual-loop reasoning: data-driven insight narrows the search space, while experience-guided generation avoids redundant or suboptimal candidates. Together, they enable more effective optimization under limited iteration budgets.

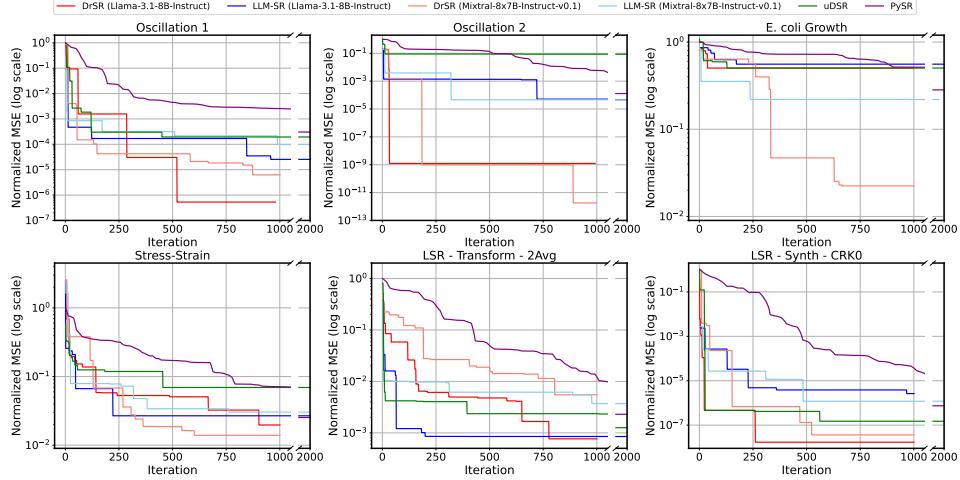


Figure 3: **Training convergence comparison.** DrSR demonstrates faster convergence and lower final error across all tasks, consistently outperforming baselines even in early-stage iterations.

### 5.4 Valid Solution Rate Analysis

To further explain the superior efficiency of DrSR, we evaluate the *valid solution rate*—the proportion of generated equations that are syntactically valid, compilable, and executable. Figure 4 compares this rate for DrSR and LLM-SR (both using Mixtral). DrSR consistently yields a higher valid rate across all tasks, especially on structurally complex datasets like E. coli Growth and Stress-Strain, where it maintains 0.4–0.6 validity compared to LLM-SR’s sub-0.3 plateau. This improvement is largely due to DrSR’s Idea Library, which captures and applies error-avoidance strategies extracted from prior failures. By reducing the frequency of invalid generations, DrSR lowers computational waste and improves sample efficiency—key advantages in practical SR under limited resources.

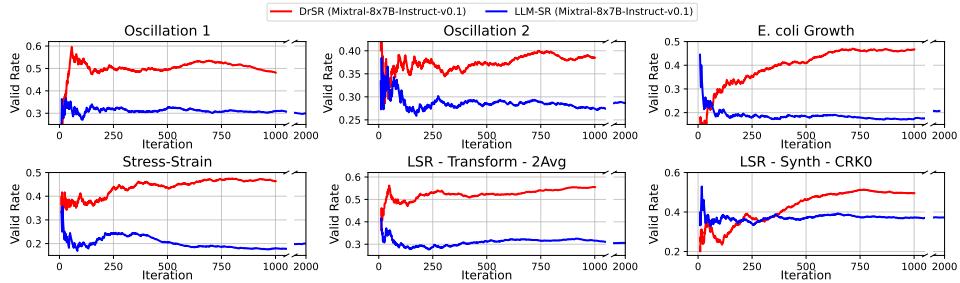


Figure 4: **Valid solution rate comparison.** We report the proportion of syntactically valid, compilable, and evaluable equations produced by DrSR and LLM-SR.

### 5.5 Ablation Study

We conduct ablation experiments on the Oscillator 2 task using Mixtral backbone to evaluate the contribution of DrSR’s two key components: *Data-aware Insight* and *Inductive Idea Extraction*.

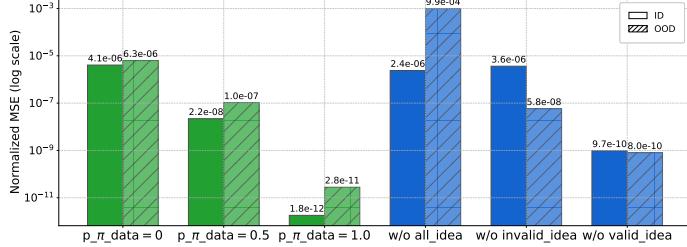


Figure 5: **Ablation results on the Oscillation 2 problem**, showing the impact of Data-aware Insight and Inductive Idea Extraction on the performance of DrSR.

**Impact of Data-aware Insight.** We vary the probability  $p \in \{0, 0.5, 1.0\}$  with which the main LLM references structured insights during generation. Setting  $p = 0$  disables data priors, while  $p = 1.0$  enforces full use. Performance improves consistently with higher usage of insights. Without data priors, the model exhibits LLM-SR-like behavior and inefficient search. In contrast, full integration ( $p = 1.0$ ) significantly accelerates convergence and improves accuracy, confirming that structural guidance sharpens the search space and enhances model stability.

**Effect of Inductive Idea Extraction.** To assess the importance of inductive feedback, we incrementally disable parts of the Inductive Idea Extraction module while keeping data insights active: *w/o valid ideas* removes strategies from successful equations; *w/o invalid ideas* also removes error-avoidance heuristics; *w/o all ideas* disables the module entirely, reducing DrSR to LLM-SR. Removing any component degrades performance, with losses compounding across settings. Valid ideas guide structural preference, while invalid ideas prevent repeat errors. Their combination enables targeted exploration, proving critical to DrSR’s performance.

## 6 Related Work

**Classical Symbolic Regression.** SR has long served as a core tool in scientific modeling [23, 24], with growing relevance in the AI4Science community [1, 16]. Early methods explored expression discovery via linear modeling, GP, evolutionary search, and reinforcement learning [25, 8, 9, 10, 26]. Sparse regression techniques [27, 28] and symbolic systems modeling [29, 30, 31, 32, 33] have extended SR to complex, nonlinear domains. GP-based approaches, initiated by Koza [34], continue to evolve with modern variants [35, 36, 37], while RL-based methods [10, 38, 39, 40, 41, 42, 43] offer adaptive equation construction under search constraints. Recently, Transformer-based models such as SymbolicGPT and SymFormer [11, 44, 12, 45, 46, 47, 48] have introduced large-scale pretraining to symbolic tasks. Despite their strong generative capabilities, these models often lack the ability to incorporate prior scientific knowledge, which is a key advantage of LLMs.

**LLMs for Symbolic Regression.** LLMs have spurred a new paradigm in SR by enabling language-driven equation generation [13, 22, 16, 49, 50]. LLM-SR [13] employs LLMs to produce equation skeletons guided by prior scientific knowledge, followed by parameter optimization. LaSR [22] extends this by incorporating abstract symbolic patterns; CoEvo [50] leverages open-ended knowledge co-evolution during search. ICSR [16] encodes training data as in-context demonstrations to generate candidate expressions. In contrast to these approaches, DrSR introduces a closed-loop reasoning framework that augments LLMs with structural insights and experience-driven reflection. By explicitly modeling the interaction between observation, generation, and feedback, DrSR enables more efficient and generalizable symbolic discovery beyond existing LLM-based variants.

## 7 Conclusion and Future Directions

We propose DrSR, a framework that augments LLM with data-driven insight and experience-guided reflection, enabling both interpretation of input data and learning from past generations. This dual reasoning design overcomes key limitations of prior LLM-based methods and delivers superior performance across diverse scientific problems, outperforming both traditional and LLM-based baselines in accuracy, generalization, and convergence. Despite its strong empirical results, DrSR also opens promising avenues for future research, including multimodal extensions to accommodate richer data types such as scientific imagery, and continual learning to accumulate transferable modeling strategies across tasks. These directions represent exciting opportunities for future exploration, and we plan to pursue them to further advance DrSR toward next-generation scientific equation discovery.

## References

- [1] Nour Makke and Sanjay Chawla. Interpretable scientific discovery with symbolic regression: a review. *Artificial Intelligence Review*, 57(1):2, 2024.
- [2] Pablo Lemos, Niall Jeffrey, Miles Cranmer, Shirley Ho, and Peter Battaglia. Rediscovering orbital mechanics with machine learning. *Machine Learning: Science and Technology*, 4(4):045002, 2023.
- [3] Benjamin L Davis and Zehao Jin. Discovery of a planar black hole mass scaling relation for spiral galaxies. *The Astrophysical Journal Letters*, 956(1):L22, 2023.
- [4] Rohit Batra, Le Song, and Rampi Ramprasad. Emerging materials intelligence ecosystems propelled by machine learning. *Nature Reviews Materials*, 6(8):655–678, 2021.
- [5] Alberto Hernandez, Adarsh Balasubramanian, Fenglin Yuan, Simon AM Mason, and Tim Mueller. Fast, accurate, and transferable many-body interatomic potentials by symbolic regression. *npj Computational Materials*, 5(1):112, 2019.
- [6] Shuo Yu, Hongyan Xue, Xiang Ao, Feiyang Pan, Jia He, Dandan Tu, and Qing He. Generating synergistic formulaic alpha collections via reinforcement learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5476–5486, 2023.
- [7] Wenyang Xu, Rundong Wang, Chen Li, Yonghong Hu, and Zhonghua Lu. Hrft: Mining high-frequency risk factor collections end-to-end via transformer. *arXiv preprint arXiv:2408.01271*, 2024.
- [8] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.
- [9] Miles Cranmer. Interpretable machine learning for science with pysr and symbolicregression. jl. *arXiv preprint arXiv:2305.01582*, 2023.
- [10] Brenden K Petersen, Mikel Landajuela, T Nathan Mundhenk, Claudio P Santiago, Soo K Kim, and Joanne T Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. *arXiv preprint arXiv:1912.04871*, 2019.
- [11] Luca Biggio, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Pascaudolo. Neural symbolic regression that scales. In *International Conference on Machine Learning*, pages 936–945. Pmlr, 2021.
- [12] Pierre-Alexandre Kamienny, Stéphane d’Ascoli, Guillaume Lample, and François Charton. End-to-end symbolic regression with transformers. *Advances in Neural Information Processing Systems*, 35:10269–10281, 2022.
- [13] Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K Reddy. Llm-sr: Scientific equation discovery via programming with large language models. *arXiv preprint arXiv:2404.18400*, 2024.
- [14] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- [15] Paul Kassianik, Baturay Saglam, Alexander Chen, Blaine Nelson, Anu Vellore, Massimo Aufiero, Fraser Burch, Dhruv Kedia, Avi Zohary, Sajana Weerawardhena, et al. Llama-3.1-foundationai-securityllm-base-8b technical report. *arXiv preprint arXiv:2504.21039*, 2025.
- [16] Matteo Merler, Katsiaryna Haitsiukevich, Nicola Dainese, and Pekka Marttinen. In-context symbolic regression: Leveraging large language models for function discovery. *arXiv preprint arXiv:2404.19094*, 2024.
- [17] Parshin Shojaee, Ngoc-Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa D Doan, and Chandan K Reddy. Llm-srbench: A new benchmark for scientific equation discovery with large language models. *arXiv preprint arXiv:2504.10415*, 2025.

- [18] Monod and J. The growth of bacterial cultures. 3(1):371–394, 1949.
- [19] L Rosso, JR Lobry, S Bajard, and Jean-Pierre Flandrois. Convenient model to describe the combined effects of temperature and ph on microbial growth. *Applied and environmental microbiology*, 61(2):610–616, 1995.
- [20] Silviu-Marian Udrescu and Max Tegmark. Ai feynman: A physics-inspired method for symbolic regression. *Science advances*, 6(16):eaay2631, 2020.
- [21] Mikel Landajuela, Chak Shing Lee, Jiachen Yang, Ruben Glatt, Claudio P Santiago, Ignacio Aravena, Terrell Mundhenk, Garrett Mulcahy, and Brenden K Petersen. A unified framework for deep symbolic regression. *Advances in Neural Information Processing Systems*, 35:33985–33998, 2022.
- [22] Arya Grayeli, Atharva Sehgal, Omar Costilla Reyes, Miles Cranmer, and Swarat Chaudhuri. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37:44678–44709, 2024.
- [23] Donald Gerwin. Information processing, data inferences, and scientific generalization. *Behavioral Science*, 19(5):314–325, 1974.
- [24] Pat Langley. Bacon: A production system that discovers empirical laws. In *IJCAI*, page 344. Citeseer, 1977.
- [25] John R Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and computing*, 4:87–112, 1994.
- [26] Fangzheng Sun, Yang Liu, Jian-Xun Wang, and Hao Sun. Symbolic physics learner: Discovering governing equations via monte carlo tree search. *arXiv preprint arXiv:2205.13134*, 2022.
- [27] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.
- [28] Kathleen Champion, Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019.
- [29] Georg Martius and Christoph H Lampert. Extrapolation and learning equations. *arXiv preprint arXiv:1610.02995*, 2016.
- [30] Subham Sahoo, Christoph Lampert, and Georg Martius. Learning equations for extrapolation and control. In *International Conference on Machine Learning*, pages 4442–4450. Pmlr, 2018.
- [31] Samuel Kim, Peter Y Lu, Sriyon Mukherjee, Michael Gilbert, Li Jing, Vladimir Čepelić, and Marin Soljačić. Integration of neural network-based symbolic regression in deep learning for scientific discovery. *IEEE transactions on neural networks and learning systems*, 32(9):4166–4177, 2020.
- [32] Min Wu, Weijun Li, Lina Yu, Wenqiang Li, Jingyi Liu, Yanjie Li, and Meilan Hao. Prunesymnet: A symbolic neural network and pruning algorithm for symbolic regression. *arXiv preprint arXiv:2401.15103*, 2024.
- [33] Yanjie Li, Weijun Li, Lina Yu, Min Wu, Jingyi Liu, Shu Wei, Yusong Deng, and Meilan Hao. Metasymnet: A tree-like symbol network with adaptive architecture and activation functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27081–27089, 2025.
- [34] John R Koza. *Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems*, volume 34. Stanford University, Department of Computer Science Stanford, CA, 1990.
- [35] Fabricio Olivetti de Franca and Guilherme Seidyo Imai Aldeia. Interaction–transformation evolutionary algorithm for symbolic regression. *Evolutionary computation*, 29(3):367–390, 2021.

- [36] Baihe He, Qiang Lu, Qingyun Yang, Jake Luo, and Zhiguang Wang. Taylor genetic programming for symbolic regression. In *Proceedings of the genetic and evolutionary computation conference*, pages 946–954, 2022.
- [37] Hengzhe Zhang, Aimin Zhou, Qi Chen, Bing Xue, and Mengjie Zhang. Sr-forest: a genetic programming based heterogeneous ensemble learning method. *IEEE Transactions on Evolutionary Computation*, 2023.
- [38] T Nathan Mundhenk, Mikel Landajuela, Ruben Glatt, Claudio P Santiago, Daniel M Faissol, and Brenden K Petersen. Symbolic regression via neural-guided genetic programming population seeding. *arXiv preprint arXiv:2111.00053*, 2021.
- [39] Mikel Landajuela, Brenden K Petersen, Soo K Kim, Claudio P Santiago, Ruben Glatt, T Nathan Mundhenk, Jacob F Pettit, and Daniel M Faissol. Improving exploration in policy gradient search: Application to symbolic optimization. *arXiv preprint arXiv:2107.09158*, 2021.
- [40] Laure Crochepierre, Lydia Boudjeloud-Assala, and Vincent Barbesant. Interactive reinforcement learning for symbolic regression from multi-format human-preference feedbacks. In *IJCAI*, pages 5900–5903, 2022.
- [41] Wenqiang Li, Weijun Li, Lina Yu, Min Wu, Linjun Sun, Jingyi Liu, Yanjie Li, Shu Wei, Yusong Deng, and Meilan Hao. A neural-guided dynamic symbolic network for exploring mathematical expressions from data. *arXiv preprint arXiv:2309.13705*, 2023.
- [42] Mengge Du, Yuntian Chen, and Dongxiao Zhang. Discover: Deep identification of symbolically concise open-form partial differential equations via enhanced reinforcement learning. *Physical Review Research*, 6(1):013182, 2024.
- [43] Jacob F Pettit, Chak Shing Lee, Jiachen Yang, Alex Ho, Daniel Faissol, Brenden Petersen, and Mikel Landajuela. Disco-dso: Coupling discrete and continuous optimization for efficient generative design in hybrid spaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27117–27125, 2025.
- [44] Mojtaba Valipour, Bowen You, Maysum Panju, and Ali Ghodsi. Symbolicgpt: A generative transformer model for symbolic regression (2021). *arXiv preprint arXiv:2106.14131*, 2021.
- [45] Martin Vastl, Jonáš Kulhánek, Jiří Kubalík, Erik Derner, and Robert Babuška. Symformer: End-to-end symbolic regression using transformer-based architecture. *IEEE Access*, 2024.
- [46] Wenqiang Li, Weijun Li, Linjun Sun, Min Wu, Lina Yu, Jingyi Liu, Yanjie Li, and Songsong Tian. Transformer-based model for symbolic regression via joint supervised learning. In *The Eleventh International Conference on Learning Representations*, 2022.
- [47] Zihan Yu, Jingtiao Ding, and Yong Li. Symbolic regression via mdlformer-guided search: from minimizing prediction error to minimizing description length. *arXiv preprint arXiv:2411.03753*, 2024.
- [48] Shizhe Ding, Boyang Xia, Milong Ren, and Dongbo Bu. Niert: Accurate numerical interpolation through unifying scattered data representations using transformer encoder. *IEEE Transactions on Knowledge and Data Engineering*, 2024.
- [49] Elliot Meyerson, Mark J Nelson, Herbie Bradley, Adam Gaier, Arash Moradi, Amy K Hoover, and Joel Lehman. Language model crossover: Variation through few-shot prompting. *ACM Transactions on Evolutionary Learning*, 4(4):1–40, 2024.
- [50] Ping Guo, Qingfu Zhang, and Xi Lin. Coeve: Continual evolution of symbolic solutions using large language models. *arXiv preprint arXiv:2412.18890*, 2024.

## Appendix

### A: Implementation and Experimental Details

#### A.1 Baseline Configurations

For gplearn, we use a symbolic regressor with a population size of 1000, tournament size of 20, and 2000 generations. The function set includes standard arithmetic and common transcendental operators (add, sub, mul, div, sin, cos, sqrt, log, abs, neg, inv). The fitness metric is mean squared error (MSE), with a parsimony coefficient of 0.01 to encourage simpler expressions. For PySR (version 0.18.2), the model is configured with a population size of 50, tournament selection size of 10, and 2000 iterations. We use standard binary operators (+, -, \*, /) and common unaries such as sin, cos, sqrt, log, and abs. For DSR, we adopt the default configuration with a batch size of 100, number of samples set to 200,000, and training noise parameter  $\text{epsilon} = 0.05$ . The symbolic operator set includes {add, sub, mul, div, sin, cos, exp, log}, and the reward metric is negative normalized MSE (neg-NMSE). We also apply entropy-based policy regularization during optimization. The uDSR configuration builds upon DSR with several extensions: the function set includes additional symbolic components such as poly and const; the prior length is capped at 20; and a polynomial optimizer is employed with degree 2 and coefficient tolerance of  $1 \times 10^{-5}$ . The training settings ( $n_{\text{samples}} = 200,000$ ,  $\text{batch\_size} = 100$ ,  $\text{epsilon} = 0.05$ ) match those used in DSR, with separate entropy and learning rate parameters adjusted for stability. For LLM-based baselines such as LLM-SR and LaSR, we use the official configurations without additional tuning or prompt modifications.

#### A.2 DrSR Configuration

DrSR retains the symbolic skeleton generation structure of LLM-SR while incorporating two additional reasoning modules. For *Data-aware Insight* ( $\pi_{\text{data}}$ ), 100 input-output pairs are sampled to estimate data structure (e.g., monotonicity, correlation), refreshed after every performance improvement. Sampling uses temperature 0.6, top- $k = 30$ , and top- $p = 0.3$ . For *Inductive Idea Extraction* ( $\pi_{\text{idea}}$ ), ideas are extracted based on categorized evaluation outcomes. At each iteration, up to three ideas are drawn from the last 50% of the existing entries per category (positive, negative, error), stored in JSON format, and reused in prompting.

#### A.3 Large Language Models

DrSR uses two backbone LLMs: LLaMA-3.1-8B-Instruct and Mixtral-8x7B-Instruct-v0.1, both quantized and deployed on NVIDIA H100 GPUs. Notably, LLaMA-3.1-8B requires only around 8GB of VRAM in 4-bit quantized form, suggesting that DrSR can also be run on less powerful machines with proper quantization. We use the same decoding parameters as LLM-SR and optimize four sampled candidate equations per iteration using BFGS. Each DrSR experiment runs for a maximum of 1000 iterations. On our setup, it takes roughly 6 hours to complete on LLaMA-3.1-8B-Instruct and 12 hours on Mixtral-8x7B-Instruct-v0.1.

## B: Dataset Descriptions

To evaluate the generality and effectiveness of DrSR, we conduct experiments on seven benchmark tasks drawn from two major sources: datasets used in the original LLM-SR study and the LLM-SRBench suite. These datasets span a variety of scientific domains, including physics, biology, chemistry, and materials science. Each dataset is associated with an underlying ground-truth expression, which serves as the target for symbolic discovery. The descriptions in Sections B.1 through B.3 are adapted from the dataset documentation in LLM-SR.

### B.1 Nonlinear Oscillators

Nonlinear damped oscillators represent a fundamental class of systems in physics and engineering, where the motion of a mass under restoring and damping forces is modeled by second-order differential equations. These systems capture the interplay between displacement, velocity, and complex external forces. Mathematically, the dynamics follow the form  $\ddot{x} + f(t, x, \dot{x}) = 0$ , where  $f$  denotes a nonlinear force term dependent on time, position, and velocity.

LLM-SR simulates two custom oscillator systems to evaluate model robustness in learning nontrivial physical relationships. Oscillator 1 is defined by:

$$\dot{v} = F \sin(\omega x) - \alpha v^3 - \beta x^3 - \gamma xv - x \cos(x) \quad (F = 0.8, \alpha = 0.5, \beta = 0.2, \gamma = 0.5, \omega = 1.0)$$

and Oscillator 2 is defined by:

$$\dot{v} = F \sin(\omega t) - \alpha v^3 - \beta xv - \delta x \exp(\gamma x) \quad (F = 0.3, \alpha = 0.5, \beta = 1.0, \delta = 5.0, \gamma = 0.5, \omega = 1.0)$$

with  $v = \dot{x}$ . Both are initialized with  $x = 0.5$ ,  $v = 0.5$ , and simulated over  $t \in [0, 50]$ . These formulations introduce strong nonlinearities and variable interactions to prevent trivial memorization and encourage structural reasoning.

These systems were chosen to highlight symbolic regression models' ability to generalize under different nonlinear dynamics and to benchmark performance beyond conventional oscillatory systems.

## B.2 Bacterial Growth

Modeling the growth dynamics of *Escherichia coli* plays a vital role in various scientific and engineering disciplines, including microbiology, biotechnology, and food safety. Accurate symbolic models that describe *E. coli* population expansion under diverse environmental conditions are critical for both scientific understanding and real-world applications.

To evaluate symbolic regression models in such contexts, LLM-SR adopt a biologically grounded yet nontrivial benchmark formulation. The population growth rate is governed by a nonlinear differential equation that incorporates four major environmental factors: population density  $B$ , substrate concentration  $S$ , temperature  $T$ , and pH level. The general form is:

$$\frac{dB}{dt} = f(B, S, T, \text{pH}) = f_B(B) \cdot f_S(S) \cdot f_T(T) \cdot f_{\text{pH}}(\text{pH})$$

where each multiplicative term controls a different aspect of bacterial behavior. To increase modeling difficulty and avoid trivial memorization, we introduce customized nonlinear designs for  $f_T(T)$  and  $f_{\text{pH}}(\text{pH})$  while maintaining realistic biological interpretations.

The full form of the growth equation used in our benchmark is:

$$\frac{dB}{dt} = \mu_{\max} B \left( \frac{S}{K_S + S} \right) \left( \frac{\tanh k(T - x_0)}{1 + c(T - x_{\text{decay}})^4} \right) \exp(-|\text{pH} - \text{pH}_{\text{opt}}|) \sin \left( \frac{(\text{pH} - \text{pH}_{\min})\pi}{\text{pH}_{\max} - \text{pH}_{\min}} \right)^2$$

This dataset presents a comprehensive challenge for symbolic solvers by integrating multiple biological priors into a structurally complex formulation. The design encourages models not only to capture meaningful interactions, but also to generalize beyond simplistic template recall.

## B.3 Material Stress Behavior

The stress-strain characteristics of materials under varying thermal conditions are fundamental to structural design and materials engineering. In this benchmark, LLM-SR adopts a real-world experimental dataset, capturing the tensile behavior of Aluminum 6061-T651 under uniaxial tension across six distinct temperatures, ranging from 20°C to 300°C. This dataset provides a practical and non-synthetic scenario, presenting significant challenges for symbolic regression models.

Unlike traditional physics-based benchmarks with known governing equations, this task does not follow a predefined theoretical model. Instead, it demands empirical discovery of complex, temperature-dependent relationships—highlighting the need for expressive modeling capacity and strong generalization ability. Notably, the symbolic expression must be inferred purely from observation without memorizing known formulas, pushing LLM-based frameworks beyond their reliance on learned priors.

To describe the temperature-dependent stress-strain relationship, one commonly used empirical formulation is:

$$\sigma = (A + B\varepsilon^n) \left( 1 - \left( \frac{T - T_r}{T_m - T_r} \right)^m \right)$$

where  $\sigma$  is the stress,  $\varepsilon$  is the strain,  $T$  is the temperature,  $T_r$  and  $T_m$  denote reference and melting temperatures, respectively, and  $A$ ,  $B$ ,  $n$ , and  $m$  are material-specific parameters.

## B.4 LSR-Transform

The LSR-Transform dataset, as introduced in LLM-SRBench, is constructed by applying symbolic transformations to problems in the Feynman benchmark. These transformations involve solving for alternative target variables and reformatting equations into mathematically valid but less familiar forms. This design prevents trivial memorization by large language models (LLMs) and emphasizes scientific reasoning and data-driven discovery.

In our DRSR experiments, we specifically selected two tasks from LSR-Transform:

**I.37.4\_0\_1** This problem involves discovering the expression for the resultant wave intensity  $I_1$  in terms of phase difference  $\delta$ , and intensities  $I_2$  and  $Int$  from two wave sources. The transformed ground-truth equation is:

$$I_1 = 2I_2 \cos^2(\delta) + I_2 + Int + 2\sqrt{I_2(I_2 \cos^2(\delta) + I_2 + Int)} \cos(\delta)$$

Here,  $I_1$  represents the resultant intensity,  $I_2$  is the intensity of the second wave source,  $Int$  is the intensity of the first source, and  $\delta$  is the phase difference.

**III.4.33\_3\_0** This problem originates from quantum harmonic oscillator systems. The target equation, with temperature  $T$  as the dependent variable, is:

$$T = \frac{h\omega}{2\pi k_b \log\left(1 + \frac{h\omega}{2\pi E_n}\right)}$$

In this context,  $T$  is the system temperature,  $E_n$  is the energy of the  $n$ th mode,  $h$  is Planck's constant,  $\omega$  is the angular frequency, and  $k_b$  is Boltzmann's constant.

These transformed tasks test the ability of LLM-based methods to reason through uncommon representations of well-known physical principles.

## B.5 LSR-Synth

The LSR-Synth dataset is designed to evaluate models' ability to discover novel equations that combine standard scientific components with synthetic terms not typically found in literature. These problems span four domains—chemistry, physics, biology, and materials science—and are validated for solvability and plausibility by human experts.

In our experiments, we included the chemistry task **CRK0**, which models reaction kinetics. The target differential equation is:

$$\frac{dA}{dt} = -0.1899 \cdot A(t)^2 + \frac{0.1899 \cdot z \cdot A(t)^2}{0.7498 \cdot A(t)^4 + 1}$$

where  $A(t)$  denotes concentration at time  $t$ , and  $dA/dt$  is the reaction rate. This equation incorporates a second-order decay term along with a synthetic nonlinear saturation term, making it an ideal test case for evaluating symbolic reasoning in data-driven kinetic modeling.

## C: Prompt

The prompts used to generate Data-aware Insights for Oscillators 1 are shown in Fig 6, where Prompt 1 corresponds to the prompt for Initial Data-aware Insight, Prompt 2 corresponds to the prompt used in Iterative Residual-based Refinement, and Prompt 3 shows the shared task requirements for Prompt 1 and 2.

The prompts used in Inductive Idea Extraction for Oscillators 1 are shown in Fig 7, corresponding respectively to the cases where the equation under analysis is Positive, Negative, and the output is Invalid.

Fig 8, 9 and 10 depict the evolution of our prompt-guided components, including *Data-aware Insight* and *Inductive Idea Extraction*, throughout the discovery process. Fig 8 corresponds to the initial insight generated at the beginning of the search. Fig 9 and 10 present the insights and ideas extracted when the model discovers its best-performing equation. These visualizations indicate that the model's *Data-aware Insight* of the dataset becomes progressively deeper, and that *Inductive Idea Extraction* are gradually refined and enriched over time.

## Prompt 1 - Initial Data-aware Insight (front part)

csv{csv\_data}

You are a data analysis expert. I have provided a dataset structure for a damped nonlinear oscillator system as follows:

The first two columns are independent variables:

x(position),

v(velocity).

The third column is the dependent variable a(acceleration).

Each row represents a set of independent variables (x, v) and their corresponding dependent variable a value.

(Task Requirements: )

## Prompt 2 - Iterative Initial Data-aware Insight (front part)

You are a data analysis expert. I will provide a dataset structure for a damped nonlinear oscillator system as follows:

previous conclusions:{last\_analysis}

dataset:{residual}

The equation corresponding to the residuals:{sample}

The first two columns are independent variables:

x(position),

v(velocity).

The third column is the dependent variable a(acceleration).

The forth column contains residuals (calculated as observed value - predicted value from the equation).

Each row represents a set of independent variables (x, v) and their corresponding dependent variable a value, and the residual value.

(Task Requirements: )

## Prompt 3 - Task Requirements for Prompt 1 & 2

Task Requirements:

1. Please help me analyze and summarize the influence of the changes in the values of different independent variables on the dependent variable, as well as the possible intrinsic relationships among different independent variables. Your response only needs to answer your analysis results in the form below, and you don't need to show your analysis process.

2.##Output Format##:

STRICLY deliver results in the following structured format:

"output\_format": {

  "analysis": {

    "independent\_to\_dependent\_relationships": {

      "x": [

        "Hint: Here you need to analyze the functional relationship between x and a in different intervals"

      ],

      "v": [

        "Hint: Here you need to analyze the functional relationship between v and a in different intervals"

    ]

```

},
"inter_relationships_between_independents": {
  "x vs v": [
    "Hint: Here you need to analyze the possible functional relationship between x
and v in different intervals. If not, you can leave it blank."
  ]
}
}

```

Figure 6: Prompt for Data-aware Insight.

#### **Prompt 4 - Inductive Idea Extraction (Positive case)**

The optimized function skeleton you just answered scored higher. Please summarize useful experience.

STRICTLY follow these rules:

1. Use the exact phrasing "when seeking for the mathematical function skeleton that represents {dependent\_name\_in\_prompt} in {problem\_name\_in\_prompt}, I can ..."
2. Summarize ONLY the key success factors
3. You need to make your answer as concise as possible

#### **Prompt 5 - Inductive Idea Extraction (Negative case)**

The optimized function skeleton you just answered scored lower. What lessons can you draw from it?

STRICTLY follow these rules:

1. Use the exact phrasing "when seeking for the mathematical function skeleton that represents {dependent\_name\_in\_prompt} in {problem\_name\_in\_prompt}, I can ..."
2. Identify ONE crucial improvement point
3. You need to make your answer as concise as possible

#### **Prompt 6 - Inductive Idea Extraction (Invalid case)**

The optimized function skeleton you just answered failed with error:

{error\_for\_sample[i]}, What lessons can you draw from it?

STRICTLY follow these rules:

1. Use the exact phrasing "when seeking for the mathematical function skeleton that represents {dependent\_name\_in\_prompt} in {problem\_name\_in\_prompt}, I need ..."
2. Address the SPECIFIC error: {error\_for\_sample[i]}
3. Identify ONE crucial improvement point
4. You need to make your answer as concise as possible

Figure 7: Prompt for Inductive Idea Extraction.

## Initial Data-aware Insight

```
### The following is the analysis about the initial data that can help you provide better answers. ###
experience 0:
# Relationships Between Independent Variables and Dependent Variable
## First independent variable (X1):
- Nonlinear relationship with Y. When X1 < -0.5, Y tends to decrease as X1 increases;
when X1 >= -0.5, Y increases
slightly with X1.
## Second independent variable (X2):
- Nonlinear relationship with Y. When X2 < -0.1, Y tends to decrease as X2 increases;
when X2 >= -0.1, Y increases
rapidly with X2.
# Intrinsic Relationships Between Independent Variables
- X1 vs X2:
- Positive correlation, correlation coefficient r=0.32 (p=0.00). When X1 increases, X2
tends to increase
simultaneously.
- Multicollinearity check: VIF=1.8 (No issue).
Note: The relationship between each independent variable and the dependent variable is
not significant (p>0.05).
```

Figure 8: Initial Data-aware Insight.

## Final Data-aware Insight

```
"output_format":
{ "analysis": {
"independent_to_dependent_relationships": { "x
": [
"The relationship between x and a is a non-linear one, with a possible cubic
relationship in the range of -0.2 to 0.2. However, the relationship becomes more
complex and less predictable outside this range.",
"The relationship between x and a is a non-linear one, with a possible
quadratic relationship in the range of -0.1 to 0.1. However, the relationship becomes
more complex and less predictable outside this range.",
"The relationship between x and a is a non-linear one, with a possible
exponential relationship in the range of -0.05 to 0.05. However, the relationship becomes
more complex and less predictable outside this range."
],
"v": [
"The relationship between v and a is a non-linear one, with a possible linear
relationship in the range of -0.5 to 0.5. However, the relationship becomes more
complex and less predictable outside this range.",
"The relationship between v and a is a non-linear one, with a possible cubic
relationship in the range of -0.3 to 0.3. However, the relationship becomes more
complex and less predictable outside this range.",
"The relationship between v and a is a non-linear one, with a possible
quadratic relationship in the range of -0.2 to 0.2. However, the relationship becomes
more complex and less predictable outside this range."
]
```

```

        ],
},
"inter_relationships_between_independents": { "x
vs v": [
    "There is a clear distinction between the cubic and quadratic relationships
between x and v, with the cubic relationship appearing more complex and
unpredictable at higher values of v."
]
}
}
}

```

Figure 9: Final Data-aware Insight.

## Final Inductive Idea Extraction

The following are previous experience summaries that can help you provide better answers.

experience 1:

When I optimize the equation framework, in order to avoid the output failure of the generated function framework, I need to ensure that the shapes of the operands in the calculations are compatible. The specific error, Execution Error: operands could not be broadcast together with shapes (1317,) (10000,), indicates that there is a size mismatch between two arrays or variables.

experience 2:

When I optimize the equation framework, in order to avoid the output failure of the generated function framework, I need to ensure that the data types of the inputs are compatible and no operations between incompatible types are performed. In this case, the specific error "Execution Error: unsupported operand type(s) for -: 'NoneType' and 'float'" indicates that the framework is trying to perform an arithmetic operation between a None value (NoneType) and a float, which is not allowed.

experience 3:

When I optimize the equation framework, in order to avoid the output failure of the generated function framework, I need to ensure that the array indices used do not exceed the bounds of the array. In this case, the error indicates that index 10 is out of bounds for an array with a size of 10. This suggests that the code is trying to access an element at an index that does not exist in the array.

experience 4:

When I optimize the equation framework, I can improve the estimation of Y for X1 and X2 values by incorporating an additional interaction parameter 'h' between x1 and x2. This change takes into account the relationship between the variables more accurately, as suggested by the residual analysis.

experience 5:

When I optimize the equation framework, I can improve the mathematical function for acceleration in a damped nonlinear oscillator by incorporating non-linear terms, interaction terms, driving forces, phase shifts, and quadratic relationships. These factors allow for a more comprehensive and accurate representation of the physical process.

experience 6:

When I optimize the equation framework, I can improve performance by considering the specific physical phenomena and balancing complexity. This is achieved through targeted adjustments for non-linear relationships, interaction, driving force, and damping factors. The key to success is a balance between the equation's structure and the physical processes it ...

experience 7:

When I optimize the equation framework, in order to avoid the generated framework having a low score, I need to identify one crucial improvement point from this failure. A potential issue could be the interaction parameter `i` between the absolute value of v and x. This parameter was introduced to capture the influence of  $|v|$  on acceleration given x, but it might not be well-defined or precise enough in its current form.

experience 8:

When I optimize the equation framework, in order to avoid the generated framework having a low score, I need to focus on one crucial improvement point from this failure. One key factor that may have caused a regression is the choice of input variables for the interaction terms. In the current version, the interaction terms are calculated using the position (\$x\$) and velocity (\$v\$) directly. However, it might be more effective to use the normalized position and velocity variables ...

experience 9:

When I optimize the equation framework, in order to avoid the generated framework having a low score, I need to focus on one crucial improvement point from this failure. One key factor that may have caused a regression is the complexity of the mathematical function. The function may have become too specific to the provided data, reducing its ability to generalize and fit other data well ...

Figure 10: Final Inductive Idea Extraction.

## D: Results Presentation

Fig 11 illustrates the normalized mean squared error (NMSE) values on the Oscillation 1 dataset. For clarity, we present the discovered equation programs in their simplified mathematical form, accompanied by optimized parameters. The program initially explores polynomial terms; as iterations progress and data understanding improves, irrelevant polynomial and constant terms are gradually eliminated. Notably, nonlinear terms—such as sinusoidal components—are identified early and consistently preserved throughout the search.

The ground-truth equation for this dataset is:

$$\dot{v} = 0.8 \sin(x) - 0.5x \cdot v - 0.5v^3 - 0.2x^3 - x \cos(x),$$

the best equation discovered after 1000 iterations by DrSR is:

$$\dot{v} = 0.8 \sin(x) - 0.5x \cdot v - 0.5v^3 - 0.2x^3$$

whereas the best equation discovered after 2000 iterations by LLM-SR is:

$$\dot{v} = -0.2x - 0.02v + 6 \times 10^{-6}x^2 - 0.03v^2 - 0.5x \cdot v + 0.08v^3 + 0.15x^3$$

The terms highlighted in pink indicate the exact components of the ground-truth equation that are correctly predicted by DrSR. This comparison highlights that DrSR captures core structural terms of the ground-truth equation, showing significantly closer alignment than LLM-SR. We attribute this improvement to DrSR's ability to integrate data-driven insight and reflective feedback in the discovery process.

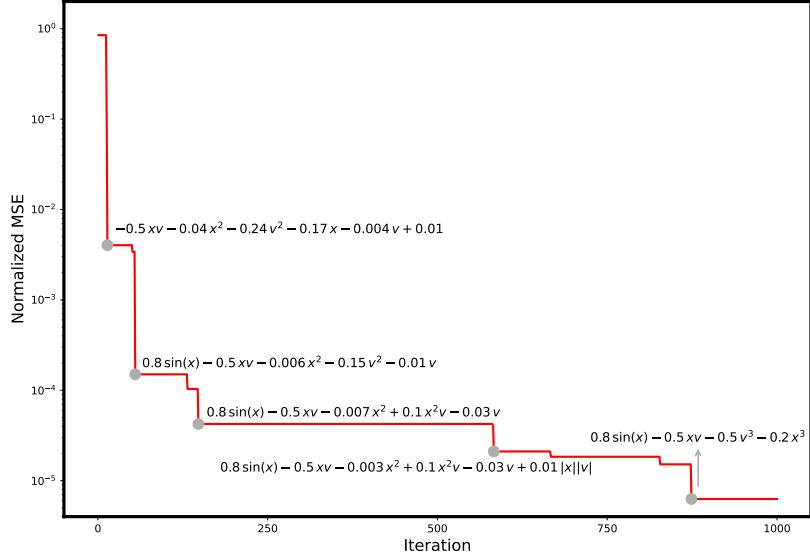


Figure 11: Performance trajectory of DrSR with the best-performing equation programs discovered over iterations on the Oscillation 1 problem.

## E: Limitation

**Limitations.** While DrSR shows strong potential for enabling LLM-guided scientific discovery, it also has several limitations. First, due to the inherent stochasticity of large language models, the generated outputs can occasionally be verbose, repetitive, or overly complex, requiring expert intervention to interpret or refine. Second, the parameter optimization phase in DRSR currently relies on the BFGS algorithm, which, while efficient, may not always achieve globally optimal parameter values for complex equation skeletons. Exploring more robust or adaptive optimization strategies remains an important direction for future work.

## F: Gaussian Noise Evaluation

To simulate the task of DrSR in real-world environments, we conducted a systematic analysis of the model’s performance under different noise levels to evaluate its robustness. We performed controlled experiments by introducing Gaussian noise with different standard deviations ( $\sigma = \{0.001, 0.002\}$ ) into the training dataset of Oscillation 1, and observed the NMSE and  $ACC_\tau$  performance on the test sets (ID/OOD).

Table 2: Comparison of DrSR and LLM-SR on Oscillator1 under different Gaussian noise levels ( $\sigma = \{0.001, 0.002\}$ ), evaluated with NMSE and accuracy under ID and OOD settings. All results use LLaMA3.1 as backbone.

Model	$\sigma = 0.001$				$\sigma = 0.002$			
	ID		OOD		ID		OOD	
	NMSE $\downarrow$	Acc $_{0.01} \uparrow$						
<b>LLM-SR</b>	1.59e-5	89.20%	0.0021	10.46%	2.68e-5	82.08%	0.0025	10.50%
<b>DrSR</b>	6.46e-6	93.24%	0.0006	31.98%	2.03e-6	95.39%	0.0011	14.46%

Table 2 presents a comparative analysis of DrSR and LLM-SR under different noise conditions. The results show that as the noise level increases, both models demonstrate noise resistance to data fluctuations. However, compared to LLM-SR, DrSR consistently achieves significantly better NMSE and  $ACC_\tau$  scores.