How Do Large Language Models Perform on PDE Discovery: A Coarse-to-fine Perspective

Xiao Luo[♡], **Changhu Wang**[♠], **Yizhou Sun**[♢] **Wei Wang**[♢],
[♡] Department of Statistics, University of Wisconsin–Madison
[♠] Department of Statistics, University of California, Los Angeles
[♢] Department of Computer Science, University of California, Los Angeles
xiao.luo@wisc.edu, wangch156@g.ucla.edu, yzsun@cs.ucla.edu, weiwang@cs.ucla.edu

Abstract

This paper studies the problem of how to use large language models (LLMs) to identify the underlying partial differential equations (PDEs) out of very limited observations of a physical system. Previous methods usually utilize physics-informed neural networks (PINNs) to learn the PDE solver and coefficient of PDEs simultaneously, which could suffer from performance degradation under extreme data scarcity. Towards this end, this paper attempts to utilize LLMs to solve this problem without further fine-tuning by proposing a novel framework named LLM for PDE Discovery (LLM4PD). The core of our LLM4PD is to utilize a coarse-to-fine paradigm to automatically discover underlying PDEs. In the coarse phase, LLM4PD selects the crucial terms from a library with hierarchical prompts and incorporates a review agent to enhance the accuracy. In the fine phase, LLM4PD interacts with a PDE solver to optimize the coefficient of the selected terms with the optimization trajectory. We also provide an adaptive hybrid optimization strategy switching between fine-tuning and exploration to balance the stability and efficiency. Extensive experiments on several systems validate the effectiveness of our LLM4PD in different settings.

1 Introduction

Partial differential equations (PDEs) have been found to drive a range of physical systems ranging from fluid dynamics (Picca, 2024) to quantum mechanics (Wu et al., 2021). However, in the real world, there are newly found complex systems where the underlying laws are unavailable (Abreu et al., 2019). Towards this end, PDE discovery (Chen et al., 2021; Stephany and Earls, 2022; Kacprzyk et al., 2024) has raised extensive interest, which aims to identify the governing PDEs of systems from the observed data.

Early data-driven approaches usually utilize shallow tools such as sparse regression (Schaeffer, 2017; Rudy et al., 2017; Gurevich et al., 2019; Berg and Nyström, 2019) to learn the coefficient of PDEs. Recently, deep learning approaches have achieved great success in this domain (Stephany and Earls, 2024a; Chen et al., 2021; Stephany and Earls, 2022), which generally utilize the power of physics-informed neural networks (PINNs) (Cai et al., 2021) to implicitly represent PDEs and their derivates. Then, they minimize the error at the observation points and the regression loss of PDE coefficients. With a strong optimizer, they can recover the governing PDEs from the data.

Despite their great success, these approaches (Stephany and Earls, 2024a; Chen et al., 2021; Stephany and Earls, 2022) usually are trained in a supervised end-to-end manner, which requires extensive observations of complex systems. However, the measurement of systems could be prohibitively expensive or even impossible in the real world and data scarcity would significantly degrade the performance (Stephany and Earls, 2024a). Towards this end, we study the problem of PDE discovery under very limited data. Recently, large language models (LLMs) have demonstrated the strong capability of few-shot learning (Song et al., 2023; Xu et al., 2024; Huang et al., 2023) and zero-shot learning (Kojima et al., 2022; Wang et al., 2024; Liu et al., 2024a; Du et al., 2024; Luo et al., 2025) without supervised fine-tuning and optimization (Yang et al., 2024a; Liu et al., 2024b). Therefore, we aim to utilize LLMs to solve the problem in this work.

In fact, incorporating LLMs into PDE discovery is a highly challenging task, which requires us to answer the following two questions. **(I)** ***How to enhance the understanding of LLMs on PDE understanding?*** LLMs are usually trained on a large amount of text data (Yang et al., 2024b), which is different from the PDE discovery problem. Due to the lack of physics knowledge, LLMs could have difficulty in understanding our problem (Polverini

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 2684–2697
November 4-9, 2025

and Gregorcic, 2024). ***(II) How to generate the coefficients of PDEs using LLMs?*** LLMs basically follow the paradigm of next-token prediction (He and Su, 2024), which could barely generate the optimal coefficients in one shot. Therefore, we need to design a holistic framework to generate the optimal coefficients gradually.

Towards this end, we propose a novel framework named LLM4PD for LLM-based PDE discovery under limited data. The high-level idea is to utilize a coarse-to-fine strategy to optimize the coefficients of PDEs. In the coarse phase, we utilize LLMs to select the crucial terms from a library of derivative terms. To enhance the understanding of LLMs on PDEs, we introduce hierarchical prompts from physics, data, and reasoning views, which can effectively guide pre-trained LLMs to align with physics knowledge. We also design a review agent to ensure the correctness of the selected terms before moving to the next phase. To utilize LLMs for effective coefficient generation, we evaluate the error between the simulation data of the current coefficients and the observations, which would be stored in a memory bank. In the fine phase, we adopt an optimization lens to follow the trajectory of the generated coefficients. To balance the stability and efficiency, we design an adaptive optimization strategy which randomly switches between the large-step exploration and small-step fine-tuning. Large-step exploration can help the model escape from local minima, while small-step fine-tuning can help the model explore the search space more efficiently. To prevent the coarse phase from generating wrong terms, we take an interleaved strategy across the coarse and fine phases based on the error reduction status. Extensive experiments on both several datasets demonstrate the effectiveness of our LLM4PD compared with competing baselines.

In summary, our contributions are three-fold: ❶ *Problem Connection*. We connect PDE discovery with LLMs and propose a novel approach to recover the underlying PDEs from the scarce data without training. ❷ *Novel Methodology*. Our LLM4PD follows a coarse-to-fine paradigm, which utilizes hierarchical prompts to reason the candidate terms and optimize the coefficients by incorporating a PDE solver and the optimization trajectory with an adaptive strategy. ❸ *Extensive Experiments*. Extensive Experiments on popular systems validate the effectiveness of our proposed LLM4PD in comparison to current state-of-the-art baselines.

2 Related Work

2.1 PDE Discovery

PDE discovery aims to identify the governing PDEs of systems from the observed data (Stephany and Earls, 2024a; Chen et al., 2021; Stephany and Earls, 2022). Early works (Schaeffer, 2017; Rudy et al., 2017; Gurevich et al., 2019) usually define a large library and utilize sparse regression to derive the coefficient of terms in PDEs. Recent deep learning approaches (Stephany and Earls, 2024a; Chen et al., 2021; Stephany and Earls, 2022) usually utilize a physics-informed neural network (PINN) (Cai et al., 2021) to represent the PDEs and minimize the error at the observation points. Then, they estimate the derivatives of the PDEs and minimize the physical loss which aligns the PDEs on the observation points with sparse regularization. However, these approaches usually require a large number of observation points to guide the recovery of PDEs (Stephany and Earls, 2024a), which could be unaffordable in practice. One recent work (Du et al., 2024) combines LLMs with sparse regression for equation discovery. In contrast, our proposed LLM4PD utilizes the capability of LLMs to recover both PDEs and coefficients from the scarce data, which could be more flexible and convenient.

2.2 LLMs for Physics

Large language models (LLMs) have shown promising results in various physics domains (Zhang et al., 2024; Liu et al., 2024c; Arlt et al., 2024) including astronomy and quantum physics. These approaches can be roughly divided into two categories. The first category aims to accelerate the physical research by providing literature tools. For example, AstroLLaMA (Nguyen et al., 2023) fine-tunes an open-source LLM using a large amount of astronomy paper. An extended version of AstroLLaMA (Perkowski et al., 2024) is also provided to help the research with interactive chatting. PhysBERT (Hellert et al., 2024) is a pre-trained language model for physics on a large corpus on over one million physics papers. The second category aims to use LLMs to solve specific physics problems. Liu et al. utilize well-trained LLMs to infer the underlying rules of Markovian transition by investigating the generation probability of LLMs (Liu et al., 2024c). In this study, our LLM4PD utilizes LLMs to discover the underlying PDEs from the observed data with extensive text-based context.

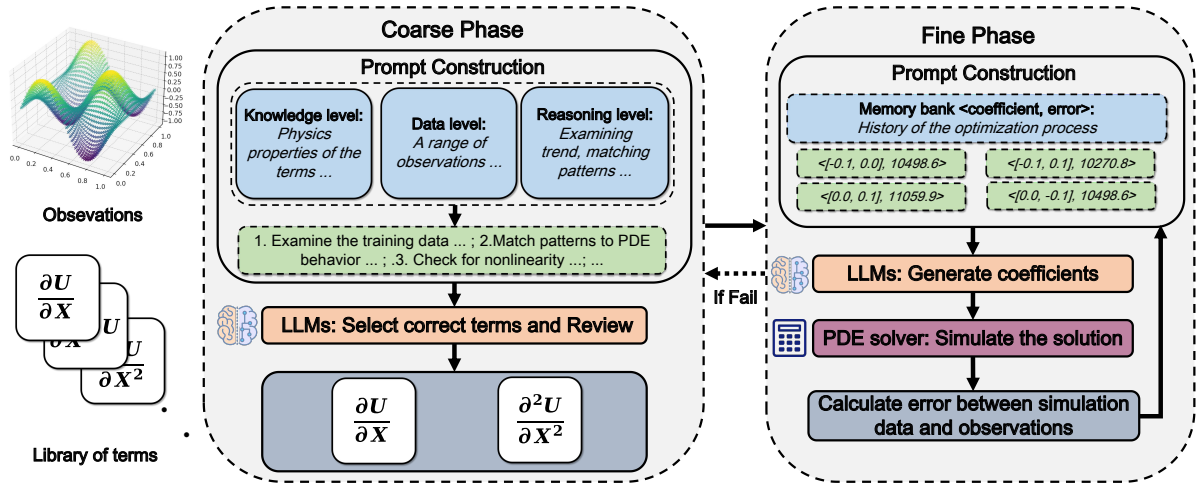

Figure 1: An Overview of LLM4PD. Our LLM4PD adopts a coarse-to-fine strategy for PDE discovery. It first utilizes hierarchical prompts to select candidate terms, followed by an LLM-based review agent, which can reduce the potential error. Then, it generates coefficients for selected terms based on the historical optimization trajectory, which can be derived from a PDE solver. If the model fails to generate proper coefficients, it will go back to the coarse phase to select the prompt terms iteratively.

3 Methodology

3.1 Problem Definition

Given a spatial domain Ω and the time interval $(0, T]$, we describe a physical system that is driven by an underlying function $u(x, t)$, i.e.,

$$u(x, t) : (0, T] \times \Omega \to \mathbb{R}, \tag{1}$$

where $x \in \Omega$ and $t \in (0, T]$. The function $u(x, t)$ is driven by an underlying PDE, i.e.,

$$\frac{\partial u}{\partial t} = \mathcal{F}(u, \theta), \tag{2}$$

where $\mathcal{F}$ is the differential operator and θ is the parameter of the PDE. Following previous works, we assume that $\mathcal{F}(u, \theta)$ is a linear combination of the basis, i.e., the derivatives of $u(x, t)$. We are given limited observation data points, $D = \{(x_i, t_i, u_i)\}_{i=1}^{n}$, where $x_i \in \Omega$, $t_i \in (0, T]$, u_i is the observation of $u(x_i, t_i)$, n is the number of the data points. We also assume there is a library of common derivative terms (Stephany and Earls, 2024a). We aim to use LLMs to automatically select the terms and estimate the coefficients.

This problem has various applications in material science, physics, and genetics. For example, it can be applied to discover the rate of gene expression change in single-cell RNA-sequencing data (Bergen et al., 2020, 2021). It can also facilitate learning governing laws of fluid dynamics (Zhang and Ma, 2020). Moreover, scientific data is usually very limited in the real world since it heavily relies on scientific experiments and measurements, which are quite costly and require extensive human effort. (Stephany and Earls, 2024a,b) also have similar observations. For example, we need expensive clinical trials to obtain single-cell transcriptome data at different periods (Liu et al., 2022). As a consequence, our studied limited data situation is quite common in reality.

3.2 Framework Overview

This paper studies the problem of discovering the underlying PDEs from very scarce observed data. Previous data-driven methods (Stephany and Earls, 2024a; Chen et al., 2021; Stephany and Earls, 2022) usually suffer from performance degradation when the observed data is limited. Here, we propose a novel approach named LLM4PD to utilize LLMs to guide a coarse-to-fine paradigm. In the coarse phase, we select the crucial terms from the whole library with extensive observation data and reasoning-based prompts in a zero-shot manner. Then, in the fine phase, we optimize the coefficients of the selected terms and validate them with the interaction with a PDE solver. The coarse and fine phases are interleaved and iteratively performed until the convergence. An overview of our LLM4PD is shown in Figure 1 and more details are provided in the following sections.

TASK: PDE Coefficient Discovery
OBJECTIVE: Find optimal coefficients [c1, c2, c3] that minimize the error for <PDE Equation>

Training Data:
First <number of observations> points (t, x, u):
<observation data>

Note: Each point is represented as (t, x, u) where:
- t: time coordinate
- x: spatial coordinate
- u: solution value at this point

Use some knowledge of the PDE to help you find the sign of the coefficients.

HISTORICAL PERFORMANCE:
Previous attempts [c1, c2, c3] -> error:
<memory bank>

BEST PERFORMANCE SO FAR:
<best performance>

TREND ANALYSIS:
1. Error Patterns:
- Track how error changes with coefficient adjustments
- Note which coefficient combinations led to lower errors
- Identify trends in successful combinations

2. Coefficient Relationships:
- Observe which coefficients tend toward zero
- Note correlations between coefficient values
- Track the impact of coefficient magnitude changes

GENERATION STRATEGY - FINE-TUNING:
Best coefficients: <best performance>
- Make SMALL adjustments (±0.1 to ±0.3) around best point
- Keep successful patterns from previous attempts
- Maintain 1-2 coefficients at zero for sparsity
- Never repeat the same coefficients from previous attempts and the current attempt

RESPONSE FORMAT:
- Provide exactly three numbers with one decimal place
- Each number must be between -1.0 and 1.0
- Separate numbers by commas
- Must be different from all previous attempts
- At least one non-zero coefficient
- Never repeat the same coefficients from the current attempt

Example: 0.5, -0.3, 0.0

Figure 2: The template example for our fine phase.

3.3 Coarse Phase for Term Selection Guided by Hybrid Contexts

The library of derivative terms is usually large and diverse (Stephany and Earls, 2024a), which makes it challenging and inefficient to find the optimal coefficients with limited data. To reduce the search space, we propose to use a reasoning-based prompt to select the crucial terms from the library. Here, we design a prompt reasoning from three different perspectives, i.e., knowledge level, data level, and reasoning level, which jointly guide the LLM to select the correct terms. We also include possible reasoning results from a review agent (Xu et al., 2023; Shi et al., 2024) in the prompt.

In particular, we begin by introducing the general information. Then, from the knowledge level, we provide the physics properties of the terms, i.e., the spatial and temporal derivatives in the library (Jiang and Luo, 2025). For example, we include that the first spatial derivative is often associated with advection, which can help LLMs align the mathematical terms with the text. From the data level, we provide a range of triplet of the observations (t, x, u), i.e., the spatial location, the time, and the value of the observation data. From the reasoning level, we provide guidance to identify the crucial terms including examining the trend of the observation data, matching patterns with PDE candidates, checking for nonlinearity, and checking from the balance of terms. With hierarchical prompts, LLMs can effectively select the crucial terms from the library.

To further reduce the error during the coarse phase, we provide an LLM-based review agent, which can examine whether the selected terms are reasonable and complete. If the proposed terms are rejected, the reason and suggestions will be incorporated into the prompt for term selection. If the review agent approves the selected terms, the process will be terminated and we will move to the fine phase.

3.4 Fine Phase for Coefficient Optimization Guided by a PDE solver

After the coarse phase, we obtain a set of crucial terms. Previous methods (Stephany and Earls, 2024a; Chen et al., 2021; Stephany and Earls, 2022) usually optimize the neural network with gradient descent or conduct regression, which could be ineffective when the training data is scarce. In contrast, we follow a heuristic paradigm to utilize LLMs for coefficient optimization, which is totally training-free and automatic process (Yang et al., 2024a; Guo et al., 2023; Liu et al., 2025). Here, we design a prompt to guide the LLM to optimize the coefficients, which consider two key points, the observation data and the optimization trajectory. Note that the optimization trajectory is the history of the optimization process, which can help LLMs to understand the optimization process and avoid the local optimal. We also design an adaptive optimization strategy switching between small-step fine-tuning and large-step exploration to improve the optimization trajectory.

In detail, to effectively optimize the coefficients with LLMs, we design a prompt with both the observation data and the optimization trajectory. Here, the observation data is included as in the previous

phase. More importantly, we utilize a PDE solver to simulate the solution of current PDEs and the error between the simulation and the observation data would be added to a memory bank $\mathcal{M}$ as follows:

$$\mathcal{M} \leftarrow \mathcal{M} \cup (< \xi_i, e_i >), \tag{3}$$

where ξ_i is the coefficient at the i-th iteration and e_i is the error between the simulation and the observation data. From the memory bank, we can monitor the optimization trajectory, which can guide the LLM in optimizing the coefficients in the next iteration. On one hand, we should pay attention to the optimization direction and update the coefficients in a small step each time to ensure the stability of the optimization. On the other hand, we should also occasionally take a large step to avoid the local optimal and enhance the efficiency of the optimization. Towards this end, we design an adaptive random optimization strategy, which can switch between small-step fine-tuning and large-step exploration based on the current error and the number of non-increasing iterations. In particular, we will select a random number r from the uniform distribution $U(0, 1)$ and compare it with a threshold. To make the optimization process more adaptive, given an error threshold E and a step length L, if the current loss e_i is above E and the current number of non-increasing iterations l_i is above L, we give a larger probability to encourage large-step exploration with a larger τ_1. Otherwise, we have τ_2. In formulation, we have:

$$\tau_i = \begin{cases} \tau_1, & \text{if } e_i > E \text{ or } l_i > L \\ \tau_2, & \text{otherwise,} \end{cases} \tag{4}$$

where τ_i is the threshold for the i-th iteration.

Then, the updating rule $\mathcal{S}_i$ is given by:

$$\mathcal{S}_i = \begin{cases} \mathcal{S}_1, & \text{if } r < \tau_i \\ \mathcal{S}_2, & \text{otherwise,} \end{cases} \tag{5}$$

where $\mathcal{S}_1$ and $\mathcal{S}_2$ are strategies for the large-step exploration and small-step fine-tuning, respectively. τ_1 and τ_2 are thresholds to determine the frequency of the small step, which is set to 0.9 and 0.5 respectively to balance the optimization process. We also provide the prompt of reasoning steps in our fine phase in Figure 2. During refinement, we restrain the output within one decimal place for simplification.

Theoretical Analysis. We present a theoretical analysis demonstrating that even when the correct PDE basis terms are selected during the coarse phase, traditional regression-based methods may still suffer from performance degradation. This highlights the necessity of the fine phase in LLM4PD. To begin, let the correct PDE basis terms be

$$\mathbf{B} = (B_1, \ldots, B_p)^\top, \tag{6}$$

where B_i represents the i-th term in the function library, such as $B_i = u\frac{\partial u}{\partial x}$. Without loss of generality, we assume that the underlying PDE is linear, given by

$$Y = \boldsymbol{\beta}^\top \mathbf{B}, \tag{7}$$

where $\boldsymbol{\beta}$ is the true coefficient vector and $Y = \frac{\partial u}{\partial t}$.

Suppose the observed dataset is denoted as $\mathcal{D}$. Using these observations, we compute the estimated evaluation $\hat{Y}$, which can be decomposed into two components: the true evaluation $\tilde{Y}$ and a bias term Y_{bias} arising from numerical differentiation over discrete observations. Mathematically,

$$\hat{Y} = \tilde{Y} + Y_{\text{bias}}. \tag{8}$$

Algorithm 1 Algorithm of LLM4PD

Input: Observation data D, library of derivative terms, pre-trained LLM.
Output: The underlying PDE.

Initialize memory bank $\mathcal{M} \leftarrow \emptyset$
while not converged **do**
  // Coarse Phase
  Generate hierarchical prompts from physics, data, and reasoning views;
  Use LLM to select crucial terms from the library;
  Review selected terms with a review agent;
  if not approved **then**
    Return to coarse phase;
  end if
  // Fine Phase
  Sample a random number $r \in U(0, 1)$;
  Generate the threshold based on the current optimization scenario using Eqn. 4;
  Generate coefficients using Eqn. 5;
  Solve the PDE with current coefficients;
  Calculate the error e_i between simulation data and observations;
  Update the memory bank $\mathcal{M}$ using Eqn. 3;
  if error not reducing for R iterations **then**
    Return to coarse phase;
  end if
end while

Task	Data	Burgers' Equation			Advection Equation			Diffusion-Reaction		
	Noise rate	0	5	10	0	5	10	0	5	10
N = 20	PDE-LEARN	0.078	0.085	0.081	0.070	**0.065**	0.101	0.077	**0.071**	0.167
	GPT-o1-mini	0.214	0.357	0.271	0.314	0.057	0.286	0.143	0.386	0.214
	LLM4PD	**0.0**	**0.0**	**0.0**	**0.0**	0.157	**0.014**	**0.029**	0.329	**0.100**
	Iter Num	68	32	59	46	-	-	-	-	
N = 40	PDE-LEARN	0.077	0.064	**0.073**	0.071	0.071	0.117	0.095	0.105	0.142
	GPT-o1-mini	0.314	0.285	0.371	0.214	0.386	0.486	0.300	0.386	0.129
	LLM4PD	**0.0**	**0.0**	0.114	**0.0**	**0.014**	**0.043**	**0.029**	**0.129**	**0.071**
	Iter Num	76	79	-	75	-	-	-	-	-
N = 60	PDE-LEARN	0.067	0.059	0.064	0.095	0.050	**0.091**	0.072	0.154	0.113
	GPT-o1-mini	0.471	0.228	0.242	0.358	0.314	0.229	0.129	0.357	0.214
	LLM4PD	**0.0**	**0.0**	**0.014**	**0.057**	**0.014**	0.242	**0.043**	**0.071**	**0.014**
	Iter Num	56	54	-	-	-	-	-	-	-

Table 1: Performance comparison of the compared models. We measure the mean absolute error (MAE) of the recovered coefficient and the ground truth following (Rudy et al., 2017). If the error is 0, it means the coefficient is exactly the same as the ground truth. *Iter Num* is the number of used up iterations if the model converges to the ground truth during the fine phase. N is the number of observation points.

Thus, the true evaluation of the observed data satisfies

$$\tilde{Y} = \boldsymbol{\beta}^\top \mathbf{B} + \varepsilon, \tag{9}$$

where ε represents the noise term. Motivated by the least squares regression framework, the coefficient vector is estimated as

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(\hat{Y}_i - \boldsymbol{\beta}^\top \mathbf{B}_i \right)^2, \tag{10}$$

where $\mathbf{B}_i$ is the estimated basis term evaluation for the i-th observation. Defining

$$\boldsymbol{\Gamma} = (\mathbf{B}_1, \ldots, \mathbf{B}_n)^\top, \quad \hat{\mathbf{Y}} = \left(\hat{Y}_1, \ldots, \hat{Y}_n \right)^\top, \tag{11}$$

the following theorem provides a theoretical analysis of the performance of regression-based methods for PDE discovery.

Theorem 3.1. *Let $\hat{\boldsymbol{\beta}}$ be decomposed as*

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + \left(\boldsymbol{\Gamma}^\top \boldsymbol{\Gamma} \right)^{-1} \boldsymbol{\Gamma}^\top \left(\boldsymbol{\varepsilon} + \mathbf{Y}_{\text{bias}} \right), \tag{12}$$

where $\boldsymbol{\varepsilon}$ is the noise term and $\mathbf{Y}_{\text{bias}}$ is the bias term.

Proof of Theorem 3.1. The proof follows directly by substituting the definition of $\hat{\boldsymbol{\beta}}$ into the least squares regression framework and expanding the terms. By definition, the least squares estimate is given by

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{\Gamma}^\top \boldsymbol{\Gamma})^{-1} \boldsymbol{\Gamma}^\top \hat{\mathbf{Y}}. \tag{13}$$

Substituting the expression for $\hat{\mathbf{Y}}$, we obtain

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{\Gamma}^\top \boldsymbol{\Gamma})^{-1} \boldsymbol{\Gamma}^\top \left(\boldsymbol{\Gamma} \boldsymbol{\beta} + \boldsymbol{\varepsilon} + \mathbf{Y}_{\text{bias}} \right). \tag{14}$$

Expanding the right-hand side, we get

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + (\boldsymbol{\Gamma}^\top \boldsymbol{\Gamma})^{-1} \boldsymbol{\Gamma}^\top \boldsymbol{\varepsilon} + (\boldsymbol{\Gamma}^\top \boldsymbol{\Gamma})^{-1} \boldsymbol{\Gamma}^\top \mathbf{Y}_{\text{bias}}. \tag{15}$$

This completes the proof. □

Theorem 3.1 highlights the impact of numerical errors introduced during derivative computation, reinforcing the necessity of the fine phase in LLM4PD. When the sample size is small, the bias term may become significant due to limited data points for estimating derivatives, resulting in $\left(\boldsymbol{\Gamma}^\top \boldsymbol{\Gamma} \right)^{-1} \boldsymbol{\Gamma}^\top \mathbf{Y}_{\text{bias}} \neq \mathbf{0}$. Similarly, when the noise is high, meaning the variance of $\boldsymbol{\varepsilon}$ is large, the term $\left(\boldsymbol{\Gamma}^\top \boldsymbol{\Gamma} \right)^{-1} \boldsymbol{\Gamma}^\top \boldsymbol{\varepsilon}$ can also be large. These findings suggest that regression-based methods may suffer from performance degradation when the sample size is small or noise levels are high, which further justifies the high challenges of our data-scarce scenarios for PDE discovery.

3.5 Interleaved Optimization

When the selected terms are not optimal, it is possible that the fine phase could fail to converge. To address this issue, we propose to interleave the coarse and fine phases. In particular, we first perform the coarse phase and then perform the fine

phase. After the fine phase, we will check whether the error can be effectively reduced. If the error fails to reduce for R iterations, we will return to the coarse phase and select the terms again. The whole algorithm is shown in Algorithm 1.

4 Experiments

4.1 Experimental Set up

Datasets. To evaluate the performance of our proposed LLM4PD, we conduct experiments on three popular PDE systems, i.e., Burgers' Equation, Advection Equation and Diffusion-Reaction Equation. Burger's Equation is widely used to model nonlinear relations and diffusion processes. Advection Equation is a basic PDE to model the transport of a scalar substance. Diffusion-Reaction Equation consists of a diffusion term and a reaction term. Following previous works (Takamoto et al., 2022; Hao et al., 2023), we construct the datasets with a high-resolution PDE solver.

Baseline and Evaluation. We compare our LLM4PD with two baselines, i.e., the original GPT model (Achiam et al., 2023) (GPT-o1-mini as default) without our adaptive design, and a state-of-the-art PDE discovery model PDE-LEARN (Stephany and Earls, 2024a). We measure the mean absolute error (MAE) of the recovered coefficient and the ground truth for each single run.

Implementation Details. We utilize GPT-o1-mini as our default model due to its strong reasoning capacity and also attempt the other LLMs in our experiments. The threshold τ_1 and τ_2 are set to 0.9 and 0.5, respectively to balance the large-step exploration and small-step fine-tuning. The total cost of the experiments for API usage is about 300 dollars.

4.2 Performance Comparison

We compare our method with two baselines, i.e., the original GPT model, and PDE-LEARN across three popular PDE systems under label scarcity in Table 1. We measure the mean absolute error of the recovered coefficient and the ground truth. If our model can recover the ground truth before using up all the iterations, we set the error to 0 and record the number of iterations used. From the results, we have the following four observations: ❶ Our proposed LLM4PD achieves the best performance in comparison with two baselines in most cases, which validates the effectiveness of our LLM4PD. The GPT model without our design is not able to recover the ground truth even when the number of training data is large, which could result from the lack of domain knowledge and interaction with a PDE solver. The performance improvement of our LLM4PD can be attributed to two factors: (i) The introduction of the coarse-to-fine design, which can focus on the crucial derivative and simplify the optimization space; (ii) The introduction of the adaptive optimization strategy, which can dynamically switch between large-step exploration and small-step fine-tuning.

❷ Our model is robust to the noise in the dataset. We can observe that the noise in the data does not affect the performance of our model largely. In comparison, PDE-LEARN is more sensitive to noise and has a higher error rate under label scarcity

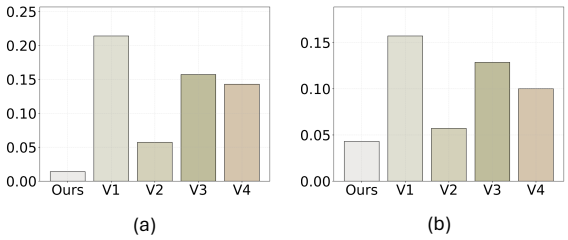

Figure 3: Ablation studies of our subcomponents in our LLM4PD with different noise rates.

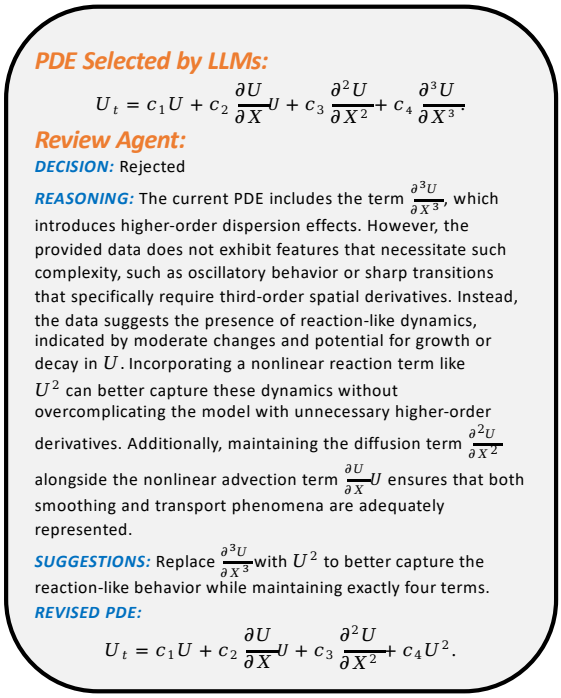

Figure 4: An example of how LLM-based review agent works. We can observe that our agent successfully revises the selections.

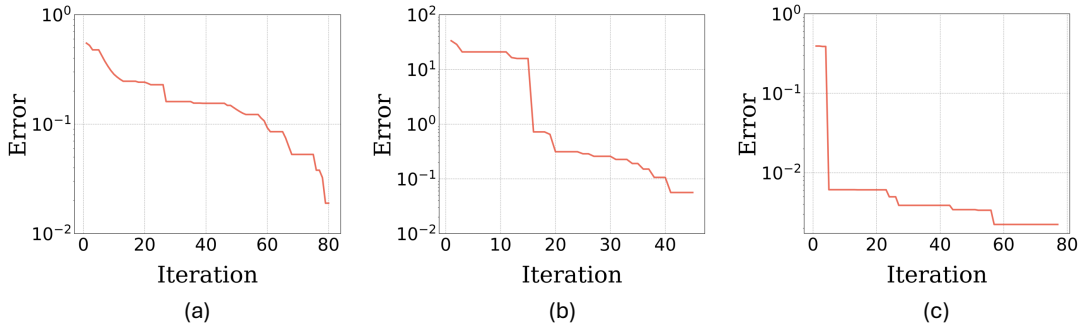

Figure 5: The observation errors between the predicted solution and the ground truth for three examples on Burgers' Equation (a) Advection Equation (b) Diffusion-Reaction (c), respectively.

as well. ❸ Our model is robust to the training data size. We can observe that with few training data the performance of our model is still more stable and effective in comparison to PDE-LEARN which is based on the physics-informed neural network. The potential reason is that our model can infer the important derivatives and coefficients using the reasoning ability of LLMs from the data without any learning process. ❹ Our proposed LLM4PD generally performs much better for simpler systems such as Advection Equation, which is evidenced by achieving the underlying ground truth during the iterations. As for Diffusion-Reaction Equation, although our proposed LLM4PD cannot find the ground truth, it still results in low errors than the compared baselines in most cases.

4.3 Further Analysis

Ablation Study. To further validate the effectiveness of the subcomponents in LLM4PD, we conduct ablation studies to analyze the impact of each critical component. In particular, we introduce the following model variants: (i) V1, which removes the review agent in the coarse phase; (ii) V2, which removes the reasoning-based context in the context for term selection; (iii) V3, which removes the memory bank in the fine phase; (iv) V4, which replaces the adaptive optimization strategy with a fixed optimization strategy. The compared MAE on the Advection Equation with noise rates of 5% and 10% can be found in Figure 3 (a) and (b) respectively. From the results, we have the following observations: ❶ Our full model outperforms V1, which validates that our review agent can help effectively avoid the wrong terms selected in the coarse phase. ❷ V2 performs much worse than our full model, which demonstrates the reasoning ability of LLMs is crucial for the success of our LLM4PD. ❸ We can observe a performance decline when removing the memory bank. The potential reason is that the memory bank provides the optimization trajectory for LLMs to generate the next guess. ❹ V4 achieves inferior performance to our full model, which indicates that our adaptive optimization strategy can effectively balance the exploration and fine-tuning for improved performance.

4.4 Further Analysis

Case Study of Our Review Agent. To further understand the performance of our proposed LLM4PD, we demonstrate the output of a review agent on the Diffusion-Reaction Equation. The results are shown in Figure 4. Note that the ground truth should include three terms i.e., U, U^2, and $\frac{\partial^2 U}{\partial x^2}$. From the results, we can observe that LLM4PD originally selects the wrong terms $\frac{\partial U}{\partial x}U$ and $\frac{\partial^3 U}{\partial x^3}$ and misses U^2. After the review agent, our LLM4PD replaces $\frac{\partial^3 U}{\partial x^3}$ with the correct term.

Visualization of Errors. To further understand the performance of our LLM4PD, we visualize the historical minimal errors of our LLM4PD during the fine phase. The results on Burgers' Equation, Advection Equation and Diffusion-Reaction Equation are shown in Figure 5. From the results, we can observe that the errors decrease smoothly during the fine phase. The potential reason is our adaptive optimization strategy can effectively switch between large-step exploration and small-step fine-tuning, which can effectively balance the stability and convergence speed. We can also observe that the decrease of the curve in Figure 5 (c) is shaper than the other curves. The reason is that the optimization trajectory is related to the characteristics of the equations. In particular, due to the introduction of UU_x, the solution of Burgers' Equation is more stable with different parameters while the advec-

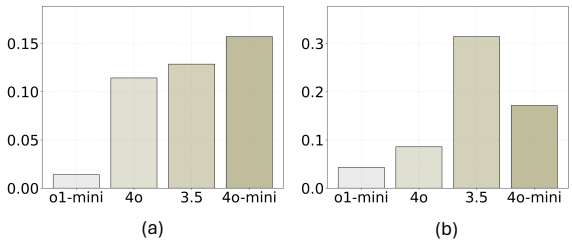

Figure 6: Performance of our proposed LLM4PD with respect to different LLMs.

tion Equation is more sensitive to the parameters, which brings in an unstable optimization curve.

Performance with respect to Different LLMs. Here, we analyze the performance of our LLM4PD w.r.t. different LLMs. We utilize four LLMs as our basic LLMs, i.e., GPT-o1-mini, GPT-4o, GPT-3.5-turbo, and GPT-4o-mini. The compared performance on the Advection Equation with noise rates of 5% and 10% can be found in Figure 6 (a) and (b), respectively. From the results, we can observe that GPT-o1-mini achieves the best performance consistently. The potential reason is that the reasoning ability and math problem-solving ability of GPT-o1-mini are the strongest among all these commercial models. Due to the resource limitation, we do not try stronger models such as GPT-o1, which would be left in our future work.

5 Conclusion

In this paper, we propose a novel framework named LLM4PD, which can effectively recover the underlying PDEs under extreme label scarcity. Our proposed LLM4PD adopts a coarse-to-fine framework. In the coarse phase, we identify the crucial terms from a library, followed by an LLM-based review agent for further checking. In the fine phase, we combine LLMs with a PDE solver, which provides feedback using the limited data source. Extensive experiments on benchmark datasets validate the superiority of the proposed LLM4PD.

6 Limitations

One limitation of this work is that we only explore the potential ability of LLMs for PDE discovery using the zero-shot setting. We believe that with further fine-tuning, LLMs could achieve better performance. Another limitation of the work is that as the pioneering work in this direction, we only include simple examples. In the future, we aim to extend our framework to handle more complex PDEs. In particular, we can incorporate both LLM-based simulators and multi-stage optimization for PDE discovery. First, we simulate the trajectories for complicated terms, and then adopt LLMs or VLMs to summarize the observations to obtain the relationships. Second, we can use multi-stage strategies which start from the generation of a subset of terms with careful validations and then select the final terms from the subset.

Acknowledgement

This work was partially supported by NSF 2211557, NSF 2119643, NSF 2303037, NSF 2312501, NSF 2200274, NSF 2106859, NIH U54HG012517, NIH U24DK097771, NIH U54OD036472, NEC, Optum AI, SRC JUMP 2.0 Center, Amazon Research Awards, and Snapchat Gifts.

References

Clare I Abreu, Jonathan Friedman, Vilhelm L Andersen Woltz, and Jeff Gore. 2019. Mortality causes universal changes in microbial community composition. *Nature communications*, 10(1):2120.

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

Sören Arlt, Haonan Duan, Felix Li, Sang Michael Xie, Yuhuai Wu, and Mario Krenn. 2024. Meta-designing quantum experiments with language models. *arXiv preprint arXiv:2406.02470*.

Jens Berg and Kaj Nyström. 2019. Data-driven discovery of pdes in complex datasets. *Journal of Computational Physics*, 384:239–252.

Volker Bergen, Marius Lange, Stefan Peidli, F Alexander Wolf, and Fabian J Theis. 2020. Generalizing rna velocity to transient cell states through dynamical modeling. *Nature biotechnology*, 38(12):1408–1414.

Volker Bergen, Ruslan A Soldatov, Peter V Kharchenko, and Fabian J Theis. 2021. Rna velocity—current challenges and future perspectives. *Molecular systems biology*, 17(8):e10282.

Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. 2021. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738.

Zhao Chen, Yang Liu, and Hao Sun. 2021. Physics-informed learning of governing equations from scarce data. *Nature communications*, 12(1):6136.

Mengge Du, Yuntian Chen, Zhongzheng Wang, Longfeng Nie, and Dongxiao Zhang. 2024. Large language models for automatic equation discovery of nonlinear dynamics. *Physics of Fluids*, 36(9).

Pei-Fu Guo, Ying-Hsuan Chen, Yun-Da Tsai, and Shou-De Lin. 2023. Towards optimizing with large language models. *arXiv preprint arXiv:2310.05204*.

Daniel R Gurevich, Patrick AK Reinbold, and Roman O Grigoriev. 2019. Robust and optimal sparse regression for nonlinear pde models. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(10).

Zhongkai Hao, Jiachen Yao, Chang Su, Hang Su, Ziao Wang, Fanzhi Lu, Zeyu Xia, Yichi Zhang, Songming Liu, Lu Lu, et al. 2023. Pinnacle: A comprehensive benchmark of physics-informed neural networks for solving pdes. *arXiv preprint arXiv:2306.08827*.

Hangfeng He and Weijie J Su. 2024. A law of next-token prediction in large language models. *arXiv preprint arXiv:2408.13442*.

Thorsten Hellert, João Montenegro, and Andrea Pollastro. 2024. Physbert: A text embedding model for physics scientific literature. *APL Machine Learning*, 2(4).

Xijie Huang, Li Lyna Zhang, Kwang-Ting Cheng, and Mao Yang. 2023. Boosting llm reasoning: Push the limits of few-shot learning with reinforced in-context pruning. *arXiv e-prints*, pages arXiv–2312.

Dapeng Jiang and Xiao Luo. 2025. Marrying llms with dynamic forecasting: A graph mixture-of-expert perspective. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 396–410.

Krzysztof Kacprzyk, Zhaozhi Qian, and Mihaela van der Schaar. 2024. D-cipher: discovery of closed-form partial differential equations. *Advances in Neural Information Processing Systems*, 36.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Weizhen Li and Rui Carvalho. 2024. Automating the discovery of partial differential equations in dynamical systems. *arXiv preprint arXiv:2404.16444*.

Fei Liu, Xi Lin, Shunyu Yao, Zhenkun Wang, Xialiang Tong, Mingxuan Yuan, and Qingfu Zhang. 2025. Large language model for multiobjective evolutionary optimization. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 178–191. Springer.

Liang Liu, Dong Zhang, Shoushan Li, Guodong Zhou, and Erik Cambria. 2024a. Two heads are better than one: Zero-shot cognitive reasoning via multi-llm knowledge fusion. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 1462–1472.

Ruishan Liu, Angela Oliveira Pisco, Emelie Braun, Sten Linnarsson, and James Zou. 2022. Dynamical systems model of rna velocity improves inference of single-cell trajectory, pseudo-time and gene regulation. *Journal of Molecular Biology*, 434(15):167606.

Shengcai Liu, Caishun Chen, Xinghua Qu, Ke Tang, and Yew-Soon Ong. 2024b. Large language models as evolutionary optimizers. In *2024 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE.

Toni JB Liu, Nicolas Boullé, Raphaël Sarfati, and Christopher J Earls. 2024c. Llms learn governing principles of dynamical systems, revealing an in-context neural scaling law. *arXiv preprint arXiv:2402.00795*.

Xiao Luo, Binqi Chen, Haixin Wang, Zhiping Xiao, Ming Zhang, and Yizhou Sun. 2025. How do large language models perform in dynamical system modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 866–880.

Tuan Dung Nguyen, Yuan-Sen Ting, Ioana Ciucă, Charlie O'Neill, Ze-Chang Sun, Maja Jabłońska, Sandor Kruk, Ernest Perkowski, Jack Miller, Jason Li, et al. 2023. Astrollama: Towards specialized foundation models in astronomy. *arXiv preprint arXiv:2309.06126*.

Ernest Perkowski, Rui Pan, Tuan Dung Nguyen, Yuan-Sen Ting, Sandor Kruk, Tong Zhang, Charlie O'Neill, Maja Jablonska, Zechang Sun, Michael J Smith, et al. 2024. Astrollama-chat: Scaling astrollama with conversational and diverse datasets. *Research Notes of the AAS*, 8(1):7.

Davide Picca. 2024. Fluid dynamics-inspired emotional analysis in Shakespearean tragedies: A novel computational linguistics methodology. In *Proceedings of the 2nd Workshop on Mathematical Natural Language Processing @ LREC-COLING 2024*, pages 11–18, Torino, Italia. ELRA and ICCL.

Giulia Polverini and Bor Gregorcic. 2024. How understanding large language models can inform the use of chatgpt in physics education. *European Journal of Physics*, 45(2):025701.

H Ribera, S Shirman, AV Nguyen, and NM Mangan. 2022. Model selection of chaotic systems from data with hidden variables using sparse data assimilation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(6).

Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. 2017. Data-driven discovery of partial differential equations. *Science advances*, 3(4):e1602614.

Hayden Schaeffer. 2017. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446.

Zhengliang Shi, Shen Gao, Xiuyi Chen, Yue Feng, Lingyong Yan, Haibo Shi, Dawei Yin, Pengjie Ren, Suzan Verberne, and Zhaochun Ren. 2024. Learning to use tools via cooperative and interactive agents. *arXiv preprint arXiv:2403.03031*.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009.

Robert Stephany and Christopher Earls. 2022. Pde-read: Human-readable partial differential equation discovery using deep learning. *Neural Networks*, 154:360–382.

Robert Stephany and Christopher Earls. 2024a. Pde-learn: Using deep learning to discover partial differential equations from noisy, limited data. *Neural Networks*, 174:106242.

Robert Stephany and Christopher Earls. 2024b. Weak-pde-learn: A weak form based approach to discovering pdes from noisy, limited data. *Journal of Computational Physics*, 506:112950.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. 2022. Pdebench: An extensive benchmark for scientific machine learning. *Advances in Neural Information Processing Systems*, 35:1596–1611.

Duo Wang, Yuan Zuo, Fengzhi Li, and Junjie Wu. 2024. Llms as zero-shot graph learners: Alignment of gnn representations with llm token embeddings. *arXiv preprint arXiv:2408.14512*.

Sixuan Wu, Jian Li, Peng Zhang, and Yue Zhang. 2021. Natural language processing meets quantum physics: A survey and categorization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3172–3182, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Derek Xu, Tong Xie, Botao Xia, Haoyu Li, Yunsheng Bai, Yizhou Sun, and Wei Wang. 2024. Does few-shot learning help llm performance in code synthesis? *arXiv preprint arXiv:2412.02906*.

Hao Xu and Dongxiao Zhang. 2021. Robust discovery of partial differential equations in complex situations. *Physical Review Research*, 3(3):033270.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. 2023. Towards reasoning in large language models via multi-agent peer review collaboration. *arXiv preprint arXiv:2311.08152*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2024a. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. 2024b. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6):1–32.

Jun Zhang and Wenjun Ma. 2020. Data-driven discovery of governing equations for fluid dynamics based on molecular simulation. *Journal of Fluid Mechanics*, 892:A5.

Yu Zhang, Xiusi Chen, Bowen Jin, Sheng Wang, Shuiwang Ji, Wei Wang, and Jiawei Han. 2024. A comprehensive survey of scientific large language models and their applications in scientific discovery. *arXiv preprint arXiv:2406.10833*.

A Details of Datasets

Burgers' Equation. Burgers' equation is widely used to model non-linear relations and diffusion processes. It is usually written as:

$$U_t + UU_x = \nu U_{xx}, \tag{16}$$

where $U = U(x,t)$ is the solution, x and t are the spatial and temporal coordinates, and ν is a hyperparameter.

Advection Equation. Advection equation is a basic PDE to model the transport of a scalar substance. The 1D Advection Equation is formulated as:

$$U_t + \nu U_x = 0 \tag{17}$$

where $U = U(x,t)$ is the solution, x and t are the spatial and temporal coordinates, and ν is a hyperparameter.

Diffusion-Reaction Equation. Diffusion-Reaction Equation consists of a diffusion term and a reaction term. In this paper, we adopt the following formulation:

$$U_t = U(1-U) + \nu U_{xx}, \tag{18}$$

where ν is a hyperparameter.

B Implementation Details

For the first two systems, we select three terms from the library. For the last terms, we select four terms from the library due to its complexity. Since LLMs could have difficulty in understanding numbers with long digits, we keep one decimal for the coefficients. We also limit the search space for every coefficient in $[-1, 1]$. The whole iteration number of our LLM4PD is set to 80 to avoid looping and save the cost. For our baseline, we strictly follow the settings in the corresponding paper.

C More Experiments

C.1 Performance Comparison with Linear Regression

Here, we introduce a model variant LLM4PD w/ LR, which replaces our fine phase with linear regression. The results are shown in Table 2. We can observe that our model outperforms LLM4PD w/ LR. The reason is that in scenarios with very limited and noisy data, the derivative cannot be calculated accurately, which validates our challenging data-scarce scenarios.

Method	20 Samples	60 Samples
LLM4PD	0.0	0.014
LLM4PD w/ LR	0.4614	0.2155

Table 2: The MAE of the compared methods on Burgers' Equation.

C.2 Performance of Review Agent

In this part, we demonstrate the times of correction for the review agent with respect to different noise rates. The compared performance is shown in Table 3. From the results, we can observe that our review agent can help effectively avoid the wrong terms selected in the coarse phase, especially under noise.

Settings	Noise rate = 0	Noise rate = 5	Noise rate = 10
Times	0	1	4

Table 3: The times of correction for the review agent with respect to different noise rates.

D Discussion

D.1 Discussion About Human Experts

Human experts have limitations in PDE discovery for two reasons. *Firstly*, human experts could struggle to process various noisy observation points simultaneously. While they can employ various analytical tools such as regression models, the derivative cannot be calculated accurately under noise, which would result in poor performance. *Secondly*, when data points are irregularly sampled, human experts face difficulties in reconstructing the underlying PDE patterns, which leads to difficulty in accurately estimating spatial derivatives and identifying governing equations. Although they cannot replace LLMs for PDE discovery, a promising direction is to explore hybrid approaches that combine human expertise with LLM capabilities.

D.2 Discussion About Problem Setting

Our problem setting follows the mainstream PDE discovery works (Stephany and Earls, 2022; Li and Carvalho, 2024; Stephany and Earls, 2024a; Xu and Zhang, 2021; Ribera et al., 2022), which always define a set of common terms to be selected. When it comes to large-term candidates, we can solve some typical PDEs with these candidate terms and leverage LLMs to summarize the relationships between observations and these terms. Since we only have

20 observation points, the size of the candidate set is set to 7 similar to previous works (Stephany and Earls, 2022; Ribera et al., 2022). Moreover, due to the length constraint, we cannot consider too large candidate sets. Note that the size is the same for all the compared methods for a fair comparison. When it comes to the large candidate set, we can divide the set into several groups with different properties and then conduct a two-stage selection strategy, which first selects the candidate groups and then identifies the final terms from the selected groups.

E Prompt of Examples

An examples of prompts is shown as in Figure 7.

TASK: PDE Coefficient Discovery
OBJECTIVE: Find optimal coefficients [c1, c2, c3] that minimize the error for D_t U = c1(U) + c2(D_x U)(U) + c3(D_x^2 U)*
Training Data:
First 20 points (t, x, u):
[(0.148, 0.281, -0.842), (0.959, 0.283, -1.029), (1.468, 0.355, 0.918), (0.248, 0.024, -0.911), (0.2, 0.146, 0.997), (1.076, 0.818, 0.806), (0.04, 0.587, -0.835), (0.238, 0.204, 0.958), (1.262, 0.202, 0.552), (0.832, 0.902, -0.741), (0.77, 0.379, 0.937), (0.537, 0.036, 0.149), (0.85, 0.343, -0.012), (1.46, 0.056, 0.564), (1.05, 0.339, -1.025), (0.027, 0.319, -0.429), (1.871, 0.459, 0.764), (0.138, 0.954, -0.9), (0.715, 0.345, 0.865), (0.03, 0.884, -0.754)]

Note: Each point is represented as (t, x, u) where:
- t: time coordinate
- x: spatial coordinate
- u: solution value at this point

Use some knowledge of the PDE to help you find the sign of the coefficients.

HISTORICAL PERFORMANCE:
Previous attempts [c1, c2, c3] -> error:
[-0.6, 0.8, 0.0] -> 0.569587(BEST)
[-0.7, 0.8, 0.0] -> 0.602267(Wrong Direction)
[0.6, -0.8, 0.4] -> 159797.187495(Wrong Direction)
[-0.6, 0.9, 0.0] -> 33370.460743(Wrong Direction)
[0.6, -0.8, 0.0] -> 1.086162(Wrong Direction)
[-0.6, 0.7, 0.0] -> 0.560422(BEST)
[0.6, 0.0, 0.0] -> 0.681910(Wrong Direction)

BEST PERFORMANCE SO FAR:
[-0.6, 0.7, 0.0] -> 0.560422

TREND ANALYSIS:
1. Error Patterns:
- Track how error changes with coefficient adjustments
- Note which coefficient combinations led to lower errors
- Identify trends in successful combinations

2. Coefficient Relationships:
- Observe which coefficients tend toward zero
- Note correlations between coefficient values
- Track the impact of coefficient magnitude changes

GENERATION STRATEGY - FINE-TUNING:
Best coefficients: [-0.6, 0.7, 0.0] -> 0.560422
- Make SMALL adjustments (±0.1 to ±0.3) around best point
- Keep successful patterns from previous attempts
- Maintain 1-2 coefficients at zero for sparsity
- Never repeat the same coefficients from previous attempts and the current attempt

RESPONSE FORMAT:
- Provide exactly three numbers with one decimal place
- Each number must be between -1.0 and 1.0
- Separate numbers by commas
- Must be different from all previous attempts
- At least one non-zero coefficient
- Never repeat the same coefficients from the current attempt

Example: 0.5, -0.3, 0.0

Figure 7: An example of prompt for the fine phase.