scientific reports

OPEN

Knowledge integration for physics-informed symbolic regression using pre-trained large language models

Bilge Taskin[1], Wenxiong Xie[1] & Teddy Lazebnik[1,2]✉

Symbolic regression (SR) has emerged as a powerful tool for automated scientific discovery, enabling the derivation of governing equations from experimental data. A growing body of work illustrates the promise of integrating domain knowledge into the SR to improve the discovered equation's generality and usefulness. Physics-informed SR (PiSR) addresses this by incorporating domain knowledge, but current methods often re- quire specialized formulations and manual feature engineering, limiting their adaptability only to domain experts. In this study, we leverage pre-trained Large Language Models (LLMs) to facilitate knowledge integration in PiSR. By harnessing the contextual understanding of LLMs trained on vast scientific literature, we aim to automate the incorporation of domain knowledge, reducing the need for manual intervention and making the process more accessible to a broader range of scientific problems. Namely, the LLM is integrated into the SR's loss function, adding a term of the LLM's evaluation of the SR's produced equation. We extensively evaluate our method using three SR algorithms (DEAP, gplearn, and PySR) and three pre-trained LLMs (Falcon, Mistral, and LLama 2) across three physical dynamics (dropping ball, simple harmonic motion, and electromagnetic wave). The results demonstrate that LLM integration consistently improves the reconstruction of physical dynamics from data, enhancing the robustness of SR models to noise and complexity. We further explore the impact of prompt engineering, finding that more informative prompts significantly improve performance.

Keywords Physics-informed symbolic regression, Automated scientific discovery, Domain knowledge integration, Prompt engineering

Scientific discovery is based on the interplay between empirical observation and theoretical modeling[1,2]. Throughout the subsequent centuries, this approach of expressing scientific knowledge through mathematical relationships has remained central to physics and other natural sciences[3–7], mathematical formulations have consistently provided the language through which we articulate physical laws[8]. These symbolic expressions serve not merely as descriptive tools but as engines of discovery[9], enabling scientists to derive new insights and make predictions beyond what has been directly observed.

The process of deriving mathematical expressions from experimental data, formally known as symbolic regression (SR), has traditionally been a labor-intensive endeavor requiring deep domain expertise and mathematical intuition[10–12]. Scientists have historically relied on a combination of theoretical understanding, creative insight, and iterative refinement to formulate equations that accurately capture observed phenomena[13,14]. In recent years, the emergence of machine learning (ML) and artificial intelligence (AI) has significantly accelerated the process of scientific knowledge extraction, particularly in domains that require complex pattern recognition and hypothesis generation[15–18]. These advantages occurred hand in hand with the exponential increase in available scientific data across all fields, including medicine[19], economics[20], and physics[21], to name a few. Despite their established benefits[22,23], current AI, in general, and ML models, in particular, are not explainable and lack the reproducibility and generalizability associated with symbolic mathematical expressions[24,25].

To address this gap, two main branches have emerged, one aims to make AI models more explainable[26–28], while the other tries to automate the SR process using computational methods[29]. Unlike traditional regression techniques, SR does not assume a predefined model structure but instead searches for the most appropriate mathematical representation using optimization techniques[30]. In physics, this process is particularly relevant as it allows for the derivation of governing equations from raw experimental or simulated data, offering a data-centric approach to understanding physical systems[31]. Indeed, SR has proven to be a valuable tool in this endeavor,

[1]Department of Computing, Jonkoping University, Jonkoping, Sweden. [2]Department of Information Science, University of Haifa, Haifa, Israel. ✉email: lazebnik.teddy@gmail.com

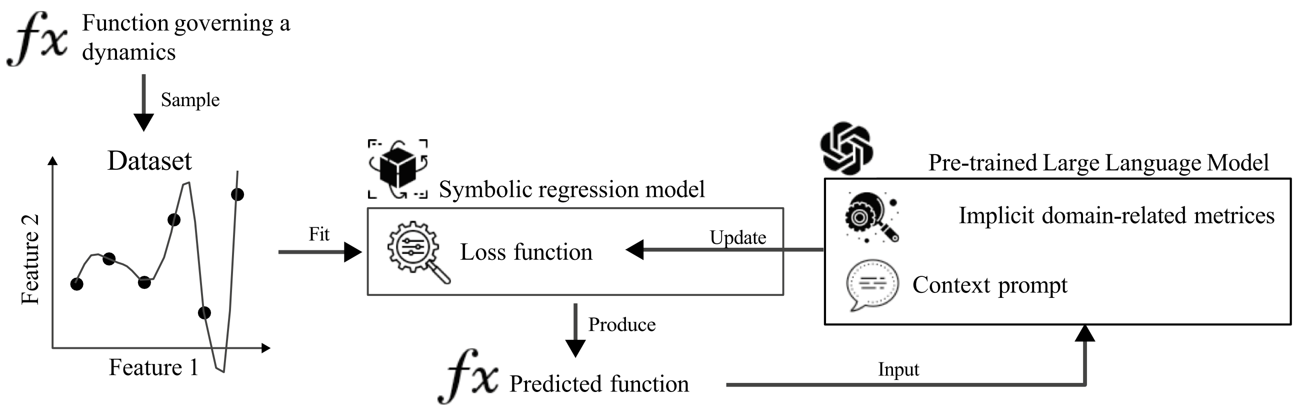
Function governing a dynamics
Sample
Dataset
Feature 2
Feature 1
Fit
Symbolic regression model
Loss function
Update
Pre-trained Large Language Model
Implicit domain-related metrices
Context prompt
Produce
Predicted function
Input

Fig. 1. A schematic view of the Large Language Model-based Knowledge Integration into the Symbolic Regression's searching (and optimization) process.

capable of rediscovering known physical laws, such as Newton's laws of motion and Maxwell's equations from observational data[32,33]. However, despite its potential, SR faces significant challenges, particularly in ensuring that the discovered expressions remain interpretable and physically meaningful rather than simply fitting a given experimental data[34–36].

In order to address these challenges, recent studies have explored physics-informed SR (PiSR), which incorporates domain knowledge to guide the discovery process[37]. These approaches leverage physical constraints, symmetries, and conservation laws to improve the quality of the inferred models. By embedding these principles into the learning process, PiSR enhances the plausibility and generalizability of discovered equations[38,39]. However, the integration of domain knowledge into PiSR remains highly technical, often requiring specialized formulations, manual feature engineering, and handcrafted constraints, making it difficult to generalize across different physical domains[37].

In this study, we propose a novel approach by leveraging pre-trained Large Language Models (LLMs), a family of deep learning models trained on vast textual and scientific datasets to develop a broad understanding of language (enabling them to generate, analyze, and integrate domain-specific knowledge in various tasks), to facilitate knowledge integration in PiSR. LLMs, trained on vast corpora of scientific literature and equations, have demonstrated remarkable capabilities in understanding and generating scientific content[40]. By harnessing the contextual understanding of LLMs, we aim to automate the incorporation of domain knowledge in SR, reducing the need for manual intervention and making the process more accessible to a broader range of scientific problems. Figure 1 shows a schematic view of the LLM-based knowledge integration into SR.

The rest of the manuscript is organized as follows. Section "Related Work"presents the current state-of-the-art in SR and LLMs, as well as previous attempts to integrate the two. Next, section "Symbolic regression" formally introduces the proposed LLM and SR integration mechanism. Following that, section "Physics-informed symbolic regression" describes the experimental setup used to evaluate the proposed method. Afterward, section "Large language models and symbolic regression" outlines the obtained results. Finally, Section "LLM-based Knowledge Integration to SR" discuss the results in an applied context and suggests possible future work.

Related work

This section provides an overview of research related to automatic scientific discovery, SR, and LLMs. We initially discussed the current state-of-the-art SR with a particular focus on PiSR, followed by a review of recent advancements in large language models, and finally we explore the integration of LLMs and SR models.

Symbolic regression

SR is a modeling technique for discovering mathematical expressions that represent the relationship between a set of input variables and an output variable[41–43]. SR does not require a predefined model structure, unlike other regression models, such as linear regression, as it searches through a large number of potential mathematical expressions to identify the function that best fits the given data[44]. This approach is particularly useful in cases where the underlying functional form is unknown or complex[45–47].

Roughly speaking, one can categorize the SR methods into four main groups based on the underlying computational techniques: brute force, sparse regression, deep learning, and genetic algorithms[48]. The brute-force-based SR methods can theoretically solves any SR tasks. Nevertheless, practically, their computational cost is unrealistic, and they are extremely prone to overfitting[49]. The method tests all possible equations to find the one that performs optimally, differing in the way they cover the search space[50]. The sparse regression SR method, which identifies parsimonious models with the help of optimization that promotes sparsity, significantly reduces the search space. Among them, SINDy is designed for scientific use cases, which employs the Lasso linear model for sparse identification of nonlinear dynamical systems behind time series data[51]. SINDy iterates between partial least squares fitting and thresholding (promoting sparsity) steps[52]. Deep learning SR methods excel in handling noisy data due to the high outlier resistance of neural networks, but are limited in their generability[53]. These methods use neural networks to produce analytical equations. For instance, Petersen et al. (2019) pro-

```
PROMPT = f"""
### ROLE
You are an expert *scientific-reasoning* assistant.
Return **ONLY** a Python-style list:
[dim_corr, simp, sim, "feedback"].

### METRICS
dim_corr . 0 (wrong) -> 1 (perfect)
simp     . 0 (complex) -> 1 (simple)
sim      . 0 (unrealistic) -> 1 (realistic)


### FEW-SHOT EXAMPLES
#1  Equation:  x = v0 * t + 0.5 * g * t^2
    Output:    [0.95, 0.80, 0.92, "Classic kinematics"]
#2  Equation:  E = m + c
    Output:    [0.05, 0.70, 0.15, "Units mismatch"]
#3  Equation:  y = sin(sin(x))
    Output:    [0.90, 0.10, 0.40, "Needless nesting"]

### TASK
Equation to evaluate:
{equation_str}

{f"Context:\n{context}" if context else ""}

Think step-by-step silently, then output the list only.
"""
```

posed a Deep Symbolic Regression (DSR) system for general SR tasks, utilizing reinforcement learning to train a generative recurrent neural network (RNN) model of symbolic expressions and employs a variant of the Monte Carlo policy gradient technique called “risk-seeking policy gradient” to adapt the generative model to the exact formulation[54]. Lastly, genetic-algorithm-based SR methods treat mathematical expressions as individuals (or genes) in a population[48]. These individuals evolve over time through mechanisms such as selection, crossover, and mutation, gradually refining the equations that can better fit the data. This evolutionary approach allows SR to discover interpretable models in a flexible way, without being constrained by predefined assumptions[55,56]. For example, the gplearn Python library implements a genetic algorithm for SR, showing a promising ability to re-discover known physical equations[57]. It first constructs a population of stochastic formulas representing the relationship between known independent variables (features) and dependent variables (objectives), presented in a tree structure. These subtrees are then replaced and restructured in a stochastic optimization process that computes fitness by executing the tree and evaluating its output, using a stochastic strategy of survival of the fittest.

In recent years, many SR algorithms have been proposed[58–60]. From these, several SR algorithms gain much popularity and provide a simple programing interface to use, including the DEAP (Distributed Evolutionary Algorithm in Python)[61,62], gplearn[63,64], and PySR[65]. All three algorithms belong to the genetic- algorithm-based SR family of methods.

Physics-informed symbolic regression
In a general sense, a learning algorithm becomes physics-informed when physical knowledge is introduced as observational, inductive, or learning biases that steer learning toward physically consistent solutions[66]. In particular, physics guidance can be implemented by designing loss functions, constraints, or inference procedures that explicitly favor solutions adhering to known physics[67,68].

SR is widely adopted for physical problems, which often face a multivariate and noisy experimental data from nonlinear systems and require analytical equations to discover general laws of nature[69–71]. In this context, simply using SR methods provides limited results as the SR needs to make sure the prediction equation follows scale consistency along with other properties[48]. To this end, Physics-Informed Symbolic Regression (PiSR) has emerged, aiming to close this gap[48,71,72]. PiSR is designed to guide the search process of symbolic expressions by incorporating physics principles such as conservation laws, symmetry, and sparsity assumptions, to name a few[71].

For example, the AI Feynman SR algorithm combines physics-inspired techniques with machine learning and deep learning methods to improve the accuracy of physics discoveries, showing promising results across multiples sub-field of physics[35]. Similarly, Martius and Lampert (2016) proposed Equation Learner (EQL) which is based on a multilayer feedforward neural network designed to learn the dynamical equations of a physical system and to be able to extrapolate to unseen domains[73]. It allows efficient training based on gradients, and its architecture consists of linear mappings, nonlinear transformations consisting of specific unitary and binary (multiplicative) units, and a linear readout layer. Golden (2024) proposed the Scalable Pruning for Rapid Identification of Null vecTors (SPRINT) algorithm, which is a model-independent sparse regression algorithm that extends the iterative singular value decomposition based exhaustive search algorithm[74].

Large language models and symbolic regression

Large language models (LLMs) have recently gained significant traction, often surpassing traditional approaches in multiple tasks in information retrieval, information representation, and natural language processing[75–80]. Research combining LLMs with established techniques such as collaborative filtering and content- based methods has yielded highly accurate recommendations[81,82]. Furthermore, LLMs are proving valuable in improving specific recommendation challenges, including sequential recommendations[83,84], long-tail recommendations[85], and cold-start scenarios[86].

Building on their success in several domains[87–90], LLMs are also being explored for their potential to enhance knowledge integration in various downstream tasks[91]. Their ability to understand and generate human-like text, coupled with their vast pre-trained knowledge, makes them promising candidates for incorporating external knowledge into models for tasks like physical symbolic regression[92]. This involves leveraging LLMs to not only understand the underlying physical principles and relationships expressed in text but also to guide the search for symbolic equations that accurately describe observed data[93]. By integrating knowledge from scientific literature, textbooks, or even expert explanations, LLMs could potentially help overcome limitations of purely data-driven symbolic regression methods, leading to more interpretable and physically plausible models.

Sharlin and Josephson (2025) show that while existing SR methods mostly rely on techniques such as genetic algorithms and Markov chain Monte Carlo sampling, they do not formally utilize available domain knowledge[94]. Even recent Transformer-based SR methods are limited in the context they are used for since they learn pattern matching between the dataset and the mathematical expression they have been provided with, limiting their view of similar problems that can be used. In order to tackle this challenge, the authors used the GPT-4 LLM (and GPT-4o) model to propose expressions based on the data, which were then optimized and evaluated using external Python tools, and then the results were fed back to the LLM to propose better expressions. Specifically, they added two hints to the LLM: One is the initial prompt, which is to input data for the model to suggest expressions; the second hint is the iterative hint, which is to combine the data and feedback for the model to optimize the expression. Merler et al. (2024) investigated the integration of LLMs into the SR process, utilizing contextual learning (ICL) capabilities[92]. The authors proposed the ICSR method, which is inspired by the Optimization by Prompting (OPRO) framework. That is, by providing LLMs with previously tested equations and fitness scores, they are made to generate better candidate equations and iterate until convergence or reach the computational budget. Using LLaMA3 8B as the underlying LLM, the ICSR method performed comparably well in all current popular SR benchmarks and generates expressions with low average complexity. Li et al. (2024) focused on how to simplify the complexity of finding expressions reflecting the relationships between variables from observed data[95]. The authors automatically generate expressions by describing requirements in natural language to an LLM. In particular, the authors proposed the MLLM-SR method, which uses LLMs to generate, run SR methods, and repeat until automatically-generated conditions are met.

In all these studies, the LLM is combined in a pipeline before or after the SR's equation prediction. In this study, we aim to integrate LLM into the SR's search process itself.

LLM-based knowledge integration to SR

In this study, we propose the integration of SR with LLMs to address to provide a physically-informed context and straightforward knowledge integration of the expert user to the SR's search process. We hypothesis that thanks to the information LLMs contain from a wide range of scientific texts they trained on, they can evaluate the equations produced by SR in terms of dimensional consistency, scientific validity, and domain knowledge. Using this capability, LLM can provide a guiding term to the SR's loss function. Notably, we do not claim that LLMs inherently understands physics nor that the score is physics-informed but rather that the LLMs can operate as simple-to-use mechanism by users to introduce domain knowledge to SR's optimization process.

To this end, let us assume an optimization system with two components, an SR and an LLM, denoted S and M, respectively. Given a dataset $D \subset \mathbb{R}^{n \times m}$ with n samples and m features, we wish to find an analytical equation that fits the data while also aligning with pre-defined list of physical constraints, represented as a metric functions accepting an equation as an expression tree and aware of the LLM's implicit knowledge. To this end, the SR component receives the dataset and a loss function L that expects a dataset, an analytical function (represented as an expression tree;[96,97]) and returns a value in $\mathbb{R}^+$ which indicates how well the analytical function fits and explains the provided dataset. Formally, we propose the following loss function:

$$L = w_1 e + w_2 s + w_3 c \tag{1}$$

where $w_1, w_2, w_3 \in [0, 1] \wedge w_1 + w_2 + w_3 = 1$, e is the mean square error[97] between the prediction of the provided analytical function and the provided dataset, s is the size of the analytical equation in terms of nodes in the

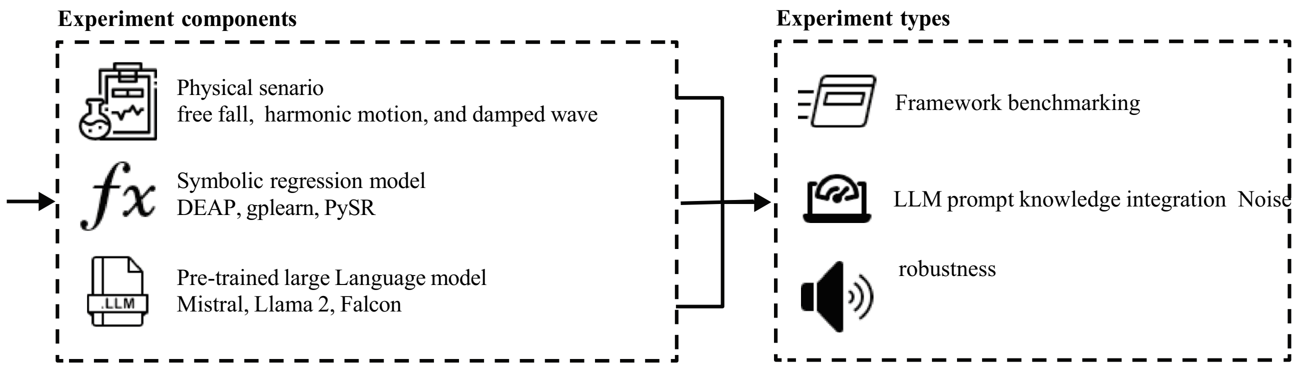
Experiment components
Physical senario
free fall, harmonic motion, and damped wave
Symbolic regression model
DEAP, gplearn, PySR
Pre-trained large Language model
Mistral, Llama 2, Falcon
.LLM
Experiment types
Framework benchmarking
LLM prompt knowledge integration Noise
robustness

Fig. 2. A schematic view of the experimental setup.

expression tree, and c is the LLM-based score of how well the analytical equation fulfills a pre-defined list of physical constraints.

Importantly, in order to ensure the reliability of the LLM-based score, which is not necessarily a floating point value but rather any string, we carefully design the LLM's prompt. To be exact, we used the following prompt:

To ensure that the prompt would work consistently across different models, we adopted a few key design principles. First, we set the temperature to zero to guarantee deterministic outputs—ensuring that models like Mistral, Llama, and Falcon would always produce the same response for the same input. We introduced a "### ROLE" section to immediately frame the assistant's focus on scientific reasoning, helping the model interpret the task correctly from the start. In the "### METRICS" section, we clearly defined three core evaluation criteria—dimensional consistency, simplicity, and physical realism—using well-spaced numerical scales so that all models could interpret them in the same way. We included a few carefully chosen examples (a classic kinematics equation, a dimensional mismatch, and an unnecessarily nested trigonometric form) to anchor the scoring logic. Finally, we enforced a strict Python-style list format for the outputs and emphasized the "ONLY" instruction to minimize the chance of extra text generation. With these measures in place, the [dim corr, simp, sim, feedback] lists returned by Mistral, Llama, and Falcon were not only structurally consistent but also reliably comparable. Importantly, the prompt included three metrics: dim corr, simp, and sim, which stand for the dimensional correction of the equation, its simplicity, and its physical realism, respectively.

To be exact, upon receiving the prompt, each LLM returns a concise, three-element list, including a score about physical dimensionality (c_1) consistency, syntactic and structural simplicity (c_2), and physical realism (c_3). The LLM's overall score is computed as follows: $c := 1 - (c_1 + c_2 + c_3)/3$.

Notably, the physics prior enters the optimization through the auxiliary term c, which evaluates whether a candidate symbolic expression respects pre-defined physical criteria. This is directly analogous to physics-informed objectives widely used across scientific machine learning, where physical knowledge is encoded as additional penalties/constraints to bias learning toward physically consistent solutions. Specifically, the LLM is prompted to return (i) dimensional consistency $(\text{dim}_c orr) and (ii) physical realism/plausibility (sim), which operationa.$

Experiments

In order to evaluate the proposed method, we designed an experiment across three in silico physical scenarios (free fall, simple harmonic motion, damped waves), using three symbolic regression implementations (PySR, DEAP-GP, gplearn) and three language models (Mistral 7B, Llama 2 7B, Falcon 7B), evaluating the robustness of the proposed method over all three axes. For each combination of these components, we conducted three experiments—benchmarking the performance, comparison between different knowledge integration provided to the LLM's prompt, and noise robustness. For all of these experiments we recorded four main metrics: mean absolute error (MAE), mean square error (MSE), coefficient of determination (R^2), and the expression tree distance. Figure 2 presents a schematic view of the experimental setup. Due to the stochastic nature of the SR algorithm, the results are shown as the mean value of $n = 25$ repetitions, balancing between computational time and statistical robustness.

Experiment components

In Silico physical experiments

As a representative set of physical problems, we adopted classical Newtonian movement cases with an increasing level of reconstruction complexity, as reflected by the number of nodes in their expression trees. All equations show non-linear dynamics, ensuring a real-world level of complexity. Namely, the three physical cases are: free fall of a ball, simple harmonic motion, and damped wave—were. First, consider a freely falling object near Earth's surface under constant gravity. When dropped from rest at height h [m], its speed follows

$$v = \sqrt{2gh}, \tag{2}$$

where $g = 9.81$ m/s^2. Next, the simple harmonic motion takes the following form:

$$x(t) = A\cos\frac{k}{m}t + \varphi, \tag{3}$$

with A as amplitude, k the spring constant, m the mass, and φ the phase offset[98]. Finally, a damped plane-wave propagation at fixed position x. The field obeys the following equation:

$$E(t) \;=\; E_0\,e^{-\alpha t/2}\cdot\cos\,kx - wt, \tag{4}$$

where α is the damping coefficient[99]. From now on, we would refer to these equations as the "ground truth" (GT) equations.

In order to generate synthetic data for each of the in silico physical scenarios, we implemented the following procedure. Initially, we conducted $N = 500$ experiments, uniformly sampling five independent parameters for their initial condition, including: (1) mass, randomly selected from 0.1 to 10 kg; (2) a characteristic length (e.g. drop height, spring amplitude, or radius) between 0.01 and 0.5 m; (3) initial height/displacement, varying from 1 to 100 m; (4) a linear drag/damping coefficient from 0 to 1 kg s^{-1}; and (5) time, sampled randomly between zero and the physically realistic maximum of $\sqrt{2h/g}$, where $g = 9.81$ m s^{-2}. Each experiment is run until reaching equilibrium or 1000 steps in time. This approach provides a broad and realistic representation of conditions[100,101].

With parameters determined, we proceeded to generate ground truth data using analytical equations corresponding to each physical scenario. These calculations provided precise, idealized responses against which the model's performance could be evaluated. For example, free fall velocity was calculated as $v = \sqrt{2gh}$ to validate the model's handling of square-root dependencies. Harmonic oscillator displacement was computed using $x = A\cos(\sqrt{k/mt} + \phi)$ to assess the model's capture of trigonometric functions. Damped wave magnitude was calculated according to $E = E_0 e^{-\alpha t/2}\cos(kx - \omega t)$ to test the model's ability to model exponential decay and oscillation. The results of these calculations served as our ground truth[32]. Importantly, these calculations occur until an equilibrium is achieved or the first time a state is observed twice (for period dynamics). On average, this process took around 100 steps in time, resulting in around 50,000 samples for each case. Recognizing the inevitability of measurement imperfections in practical settings, we added zero-mean Gaussian noise to our ideal signals. The noise level was set to 1% of each signal's variability, if not stated otherwise, resulting in an average signal-to-noise ratio (SNR) of approximately 40 dB, a value representative of typical laboratory instrumentation. This process yielded noisy observations, denoted by y, which incorporate a realistic level of measurement uncertainty[102,103]. With the five independent parameters defined, we constructed a predictor matrix X. The corresponding noisy signals were assigned as the target vector y. The data were retained in their original SI units, relying on the regression models to manage any required scaling, normalization, or transformations internally[100].

Large language models

We used three open-source and locally-depolyable LLMs—Mistral 7B, Llama 2 7B, and Falcon 7B. We picked these models due to their relative small size, which allows to run them locally on standard computer system on the one hand and their reasoning and robust training process, on the other hand. While other LLMs such as Gemini 2 and GPT-o1 that requires API (Application Programming Interface) calls may produce better results, we decided to use local LLMs as they both cheaper and ensures sensitive data is not shared with 3rd party in practical usage[104–106]. Importantly, we set the sampling temperature to zero to guarantee deterministic, repeatable outputs and to ensure that differences in plausibility scores arise solely from model knowledge rather than random sampling noise[107–109].

Symbolic regression models

We used three popular SR models: DEAP, PySR, and gplearn. A uniform population size of 100 and 50 generations (iterations) are used for all three SR models to balance between a robust search process and computational resources. Specifically, for PySR, we used the Huber loss function and enabled a complexity penalty that softly discourages extra operators, yielding more concise formulas. For DEAP, we controlled genetic variation with a crossover probability of 0.05, a mutation probability of 0.01, a tournament selection size of 3, and a maximum tree depth of 8 to prevent bloat. Finally, for gplearn, we applied a substantial complexity penalty on operator counts so that only genuinely informative structures were preserved. Across all three SR models, we utilized the following operator set: $+, -, \times, \div$, exp, log, sin, and cos. Table 1 summarizes the used hyperparameters and their values for these models.

Notably, to prevent overfitting and improve computational efficiency, we used an early-stopping rule. Optimization stopped if the best composite loss improved by less than 0.1% over three generations[110].

Experiment types

Benchmarking

For the benchmarking of the proposed LLM-integrated SR framework, we computed all 27 (3 LLMs, 3 SR, and 3 physical scenarios) configuration and compared them to the baseline configuration of each SR model in which no LLM model is used.

LLM prompt knowledge integration experiment

Due to the fact that the exact prompt used to query the LLM response can produce highly different re- sponses, and therefore numerical values for the SR optimization process, we explored the influence of different types of

Category	Hyperparameter	DEAP	PySR	gplearn
General	Population size	100	100	100
	Iterations / Generations	50	50	50
Genetic settings	Crossover probability	0.05	—	—
	Mutation probability	0.01	—	—
	Tournament size	3	—	—
Tree Settings	Tree–height limit	8	—	—
	Initial tree depth	1	—	—
Operators	Binary operators	+, −, *, /, protected div	+, −, *, /, ∧	add, sub, mul, div
	Unary operators	neg, log, sin, cos	neg, exp, log, sin, cos	neg, exp, log, sin, cos

Table 1. Hyperparameter settings for symbolic regression tools.

knowledge in prompts on the LLM's influence on the SR's produced equation. To this end, we consider eight configurations. Formally, prompt A provided no additional information, acting as a baseline. Prompt B included data-column descriptions only, testing if this reminder of physical quantities improved dimensional reasoning. Prompt C added a concise, natural-language "Experiment description", exploring whether a broader physical context improved plausibility judgments[111]. Prompt D appended the GT equation, investigating whether this reiteration of the ground truth results in the SR optimization process to converge to it[112]. In addition to the single-cue prompts, we explored combinations to investigate synergistic or competing effects. Prompt E merged B and C to assess whether pairing raw data details with contextual framing yields improved performance. Prompt F merged B and D to test whether coupling data reminders with the true formula enhances dimensional correction, potentially at the risk of over-guiding the LLM[113]. Prompt G merged C and D to evaluate if contextual description combined with ground-truth disclosure facilitates reconciling intuition with explicit answer keys[114]. Finally, Prompt H fused B, C, and D, creating a "kitchen-sink" template to determine whether information overload hinders performance or if redundancy reinforces consistent scoring[115]. The prompt variant (A–H) used for the three physical scenarios is provided as supplementary material.

Noise robustness experiment
To assess how our LLM-integrated SR model holds up when measurements get noisy, we followed the experiment design proposed by[37]. Namely, we first followed the regular data generation procedure (see section "In silico physical experiments") but without adding noise. We then added zero-mean Gaussian noise at five levels— 1%, 2%, 3%, 4%, and 5% of each signal's standard deviation—applying it separately to the input features and to the target variable.

Expression tree distance metric
Evaluating equations only with numerical error values (e.g., mean square error or coefficient of determination) answers merely the question of "how well they fit the data". However, our goal is to determine how close the discovered formulas are to the data numerically and how structurally similar they are to the real underlying formula. For this purpose, we use the *expression tree distance* method to compare each equation we obtain with the GT equations[116,117].

We compute the expression tree distance by first converting equations to expression trees[53], where operations are arranged hierarchically (e.g., "+", "-", "*"). Internal nodes represent operators with left-/right subtrees, while leaf nodes contain numbers or variables. The distance is measured by simultaneously comparing the trees from the root. Identical leaf node contents yield a distance of zero; differing variables yield one. For differing numbers, the distance is $\alpha|v_1 - v_2|$ for $\alpha \in [0, 1]$, capped at one. A leaf/internal node mismatch yields a distance of one[118]. Formally, the expression tree distance is computed as follows: mismatched operators at matched internal nodes contribute a distance of one. For identical, commutative operators (e.g., "+" or "*"), the subtree matches with the minimum total distance (direct or cross) is selected. For non-commutative operators (e.g. "-", "/"), subtree matches are strictly left-to-left and right-to- right, and subdistances are summed. These local distances are summed and normalized by the number of node comparisons, yielding a distance in [0, 1], and aggregated from root to leaves. A zero distance signifies structural equivalence (accounting for commutativity); higher values indicate structural deviations.

Results
Table 5 in supplementary material presents an evaluation of the LLM-integrated SR framework's performance, in terms of MAE, MSE, $1 - R^2$, and expression tree distance, across the three distinct physical experiments. Across all experiments and SR algorithms, Mistral generally achieves the best results followed by LLaMA, and then Falcon. Moreover, PySR consistently outperforms the other two SR models, often achieving the best performance. DEAP also tends to perform respectably, while gplearn often exhibits slightly weaker performance.

Table 2 presents the average performance of each group of LLM integrated SR models under eight different prompt variants in three sets of physical experiments in terms of MAE, MSE, $1 - R^2$, and expression tree distance. Notably, the richer the information provided to the LLM, the better the performance of the SR model. Of all the LMM models, Mistral achieves the best results, followed by LLaMA, then Falcon. In addition, PySR outperforms the other two SR models. Importantly, for the cases where the GT equation is provided to the LLM (Cases D, F, G, and H), an expression tree distance of 1.00 is obtained, meaning a perfect reconstruction of the GT equation. Similarly, the same results are obtained when the variable descriptions, together with the experiment description, are provided. While the expression tree distance is 1.00, the other fitness metrics are not identical (despite the fact that the predicted equation is identical), as each experiment had slightly different samples, which cause small fluctuations.

Table 3 summarizes the results of the noise experiment, dividing between noise addition to the source features, the target feature, for the cases of 1% to 5% standard deviation noise addition. On average, for the DEAP SR algorithm, there is an increase of the expression tree distance from 0.07 at 1% noise to 0.18 at 5% noise. In contrast, *DEAP* and *gplearn* start at lower baselines – 0.80 and 0.90 at 1% noise level and present an even steeper increase, especially under combined noise, where their expression tree distances are as small as 0.60 to 0.40 at the 5% noise level.

Table 4 summarizes the results of the biased (non-zero mean) sensitivity analysis, where we perturb the data using $\varepsilon \sim \mathrm{N}\,(\mu, \sigma^2)$ with $\mu = 1\%$ of the mean value of each feature of the data), and report performance for 1% to 5% noise levels, divided between noise added to the source features, the target variable, and both. As expected, biased noise induces a stronger degradation than the mean-zero setting: across all three experiments, the average expression-tree distance for *DEAP* increases from 0.16 at 1% feature noise to 0.32 at 5% feature noise, while under combined noise it increases from 0.44 (1%) to 0.60 (5%). A similar trend is observed for *gplearn*, which shows a steep increase under combined noise, reaching distances around 0.53–0.73 at the 5% level depending on the experiment. In contrast, *PySR* remains comparatively more robust under biased noise, with lower distances across all noise types and levels (e.g., 0.10 to 0.23 on average for feature noise from 1 to 5%), although it also degrades noticeably under combined noise at higher noise levels.

Discussion and conclusion

In this study, we proposed a method of integrating LLM with SR for physical cases. The method introduces an LLM-based score for the SR loss function, allowing a simple method of introducing physical knowledge to the SR's search process. To this end, we conducted a robust evaluation of the method using three SR algorithms (DEAP, gplearn, and PySR) together with three pre-trained LLM models (DEAP, Falcon, and Mistral) on three physical dynamics (dropping ball, simple harmonic motion, and electromagnetic wave). For each combination of these three, we computed the fitting metrics in terms of MAE, MSE, and $1 - R^2$ as well as the expression tree distance between the GT and the obtained equations.

Our results suggest that augmenting SR with an LLM-based evaluation term can improve reconstruction quality in the three physical scenarios and experimental settings studied, as indicated in Table 5 in supplementary material. The improvements are not uniform: they depend on the SR engine, the chosen LLM, and the prompt variant. We therefore interpret the LLM term as a configurable prior/regularizer that can help guide search, rather than as a guarantee of consistent gains or inherently more stable optimization across SR methods. This out- come aligns with previous studies that LLM tutors improve other data-driven model's performance[119,120]. Specifically, we find that the improvement in the SR models' performance is consistent, indicating that a reasonable usage of the LLM in the loss function, alongside the data-fitting terms results in a relatively stable optimization task[121]. Moreover, Mistral outperforms the other LLMs, which can be attributed to its comparably large size.

Moreover, exploring how different information about the experiment influences the performance of the LLM-integrated SR method, we reproduced the expected result that more information, on average, improves the method's performance, as indicated by Table 3. This outcome is in line with recent studies showing that prompt engineering can function as a powerful inductive bias for downstream models[122,123]. Specifically, our prompt-sensitivity study indicates that providing clearer variable definitions and experiment context often correlates with improved performance of the LLM-integrated SR variants. This pattern is consistent with the view that prompts can act as an inductive bias for the LLM scoring term[124]. However, we do not directly measure optimization stability, so we avoid mechanistic claims about "stabilization" and instead describe the effect as a bias toward equations the LLM rates as more plausible under the provided context. Furthermore, when evaluated on our synthetic benchmarks with known ground truth, improvements are sometimes reflected in greater structural similarity to the GT equation; we treat this as evidence of improved recovery in these controlled settings, not as a general claim of "accurate discovery" of physical dynamics in real-world data.

Furthermore, we obtained the results of three physical experiments using data in which noise was de- liberately introduced to the target variable, following the approach of previous studies[35]. In real-world scenarios, noise can affect the target variable, the input variables, or both. While prior work has mostly focused on target noise, the latter two cases, though more prevalent in practice, are more challenging for SR, as noise in the input variables can compound the effects of target noise. Our noise analysis revealed two key findings, as revealed by Tables 3 and 4. First, in the synthetic noise experiments (Gaussian noise at the tested levels), LLM-integrated variants often degrade less than their corresponding non-LLM baselines, suggesting a potential robustness benefit under the specific noise model and conditions studied. Second, the method's sensitivity to noise increases as the complexity of the underlying equation grows for all SR models (in terms of the size of the expression tree of the GT equation). Importantly, while LLM increases the computation time, its relative cost remains moderate and manageable. Specifically, the LLM term in- creases computation time, and the relative overhead is configuration-dependent: it is more modest when SR iterations are computationally heavy, but can be substantial when the SR loop is fast. In our case, the LLM-to-SR runtime ratio typically remains below 0.4, indicating that

Idx	Prompt	LLM	SR	MAE	MSE	$1-R^2$	Expression tree distance
A	No context	LLaMA	DEAP	0.120	0.030	0.08	0.05
			PySR	0.100	0.020	0.06	0.02
			gplearn	0.150	0.040	0.10	0.12
		Falcon	DEAP	0.140	0.040	0.10	0.10
			PySR	0.120	0.030	0.08	0.07
			gplearn	0.170	0.050	0.12	0.15
		Mistral	DEAP	0.110	0.030	0.07	0.03
			PySR	0.090	0.020	0.05	0.01
			gplearn	0.140	0.040	0.09	0.11
B	Variable descriptions	LLaMA	DEAP	0.110	0.028	0.07	0.04
			PySR	0.090	0.018	0.05	0.01
			gplearn	0.140	0.038	0.09	0.11
		Falcon	DEAP	0.130	0.038	0.09	0.08
			PySR	0.110	0.028	0.07	0.05
			gplearn	0.160	0.048	0.11	0.14
		Mistral	DEAP	0.100	0.028	0.06	0.02
			PySR	0.080	0.018	0.04	0.01
			gplearn	0.130	0.038	0.08	0.10
C	Experiment description	LLaMA	DEAP	0.100	0.025	0.06	0.03
			PySR	0.080	0.016	0.04	0.00
			gplearn	0.130	0.035	0.08	0.09
		Falcon	DEAP	0.120	0.035	0.08	0.06
			PySR	0.100	0.025	0.06	0.03
			gplearn	0.150	0.045	0.10	0.13
		Mistral	DEAP	0.090	0.025	0.05	0.01
			PySR	0.070	0.016	0.03	0.00
			gplearn	0.120	0.035	0.07	0.08
D	Formula at end	LLaMA	DEAP	0.060	0.008	0.03	0.01
			PySR	0.040	0.005	0.02	0.00
			gplearn	0.090	0.012	0.05	0.06
		Falcon	DEAP	0.080	0.010	0.04	0.02
			PySR	0.060	0.007	0.03	0.01
			gplearn	0.110	0.015	0.06	0.10
		Mistral	DEAP	0.050	0.006	0.02	0.00
			PySR	0.030	0.004	0.01	0.00
			gplearn	0.080	0.010	0.04	0.04
E	B + C	LLaMA	DEAP	0.040	0.004	0.01	0.00
			PySR	0.020	0.002	0.01	0.00
			gplearn	0.070	0.007	0.03	0.00
		Falcon	DEAP	0.060	0.006	0.02	0.00
			PySR	0.040	0.004	0.01	0.00
			gplearn	0.090	0.010	0.04	0.00
		Mistral	DEAP	0.030	0.003	0.01	0.00
			PySR	0.010	0.001	0.00	0.00
			gplearn	0.060	0.006	0.02	0.00
F	B + D	LLaMA	DEAP	0.040	0.004	0.01	0.00
			PySR	0.020	0.002	0.01	0.00
			gplearn	0.070	0.007	0.03	0.00
		Falcon	DEAP	0.060	0.006	0.02	0.00
			PySR	0.040	0.004	0.01	0.00
			gplearn	0.090	0.010	0.04	0.00
		Mistral	DEAP	0.030	0.003	0.01	0.00
			PySR	0.010	0.001	0.00	0.00
			gplearn	0.060	0.006	0.02	0.00
Continued							

Idx	Prompt	LLM	SR	MAE	MSE	$1-R^2$	Expression tree distance
G	C + D	LLaMA	DEAP	0.040	0.004	0.01	0.00
			PySR	0.020	0.002	0.01	0.00
			gplearn	0.070	0.007	0.03	0.00
		Falcon	DEAP	0.060	0.006	0.02	0.00
			PySR	0.040	0.004	0.01	0.00
			gplearn	0.090	0.010	0.04	0.00
		Mistral	DEAP	0.030	0.003	0.01	0.00
			PySR	0.010	0.001	0.00	0.00
			gplearn	0.060	0.006	0.02	0.00
H	B + C + D	LLaMA	DEAP	0.040	0.004	0.01	0.00
			PySR	0.020	0.002	0.01	0.00
			gplearn	0.070	0.007	0.03	0.00
		Falcon	DEAP	0.060	0.006	0.02	0.00
			PySR	0.040	0.004	0.01	0.00
			gplearn	0.090	0.010	0.04	0.00
		Mistral	DEAP	0.030	0.003	0.01	0.00
			PySR	0.010	0.001	0.00	0.00
			gplearn	0.060	0.006	0.02	0.00

Table 2. Prompt-sensitivity: MAE, MSE, $1-R^2$ and expression-tree score for each LLM–SR pair under eight prompt variants.

		Feature noise					Target noise					Both noise				
Experiment	SR Model	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%
Dropping ball	DEAP	0.17	0.22	0.26	0.29	0.30	0.29	0.37	0.40	0.42	0.44	0.46	0.52	0.55	0.56	0.57
	PySR	0.11	0.15	0.20	0.22	0.25	0.15	0.20	0.25	0.27	0.30	0.25	0.30	0.35	0.37	0.40
	gplearn	0.20	0.25	0.30	0.32	0.35	0.31	0.40	0.42	0.43	0.45	0.48	0.54	0.57	0.58	0.60
Simple harmonic motion	DEAP	0.12	0.17	0.22	0.24	0.25	0.22	0.27	0.30	0.33	0.35	0.29	0.34	0.37	0.39	0.40
	PySR	0.07	0.10	0.12	0.14	0.15	0.10	0.14	0.17	0.20	0.22	0.20	0.24	0.27	0.28	0.30
	gplearn	0.10	0.15	0.17	0.19	0.20	0.20	0.24	0.27	0.30	0.33	0.28	0.32	0.35	0.37	0.40
Electromagnetic wave	DEAP	0.10	0.15	0.19	0.20	0.22	0.25	0.30	0.32	0.34	0.35	0.35	0.40	0.42	0.44	0.45
	PySR	0.05	0.08	0.10	0.11	0.12	0.12	0.17	0.20	0.22	0.25	0.22	0.27	0.29	0.30	0.30
	gplearn	0.08	0.12	0.15	0.17	0.18	0.22	0.27	0.29	0.30	0.32	0.32	0.37	0.39	0.40	0.40

Table 3. Robustness evaluation of SR models under different noise types and levels (1%–5%) for the expression- tree score. *Note:* Values are expression-tree scores.

		Feature noise					Target noise					Both noise				
Experiment	SR Model	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%
Dropping ball	DEAP	0.20	0.26	0.31	0.35	0.37	0.36	0.45	0.49	0.52	0.54	0.55	0.62	0.66	0.68	0.70
	PySR	0.14	0.19	0.24	0.27	0.31	0.20	0.27	0.32	0.35	0.39	0.32	0.38	0.44	0.47	0.51
	gplearn	0.24	0.30	0.35	0.38	0.42	0.39	0.48	0.52	0.54	0.57	0.57	0.64	0.68	0.70	0.73
Simple harmonic motion	DEAP	0.15	0.21	0.27	0.30	0.32	0.28	0.34	0.38	0.41	0.44	0.36	0.42	0.46	0.49	0.52
	PySR	0.09	0.13	0.16	0.19	0.21	0.14	0.19	0.23	0.27	0.30	0.25	0.30	0.34	0.37	0.40
	gplearn	0.13	0.18	0.22	0.24	0.27	0.25	0.30	0.35	0.39	0.43	0.34	0.40	0.45	0.49	0.53
Electromagnetic wave	DEAP	0.13	0.19	0.24	0.26	0.28	0.31	0.37	0.40	0.43	0.46	0.42	0.48	0.52	0.55	0.58
	PySR	0.07	0.11	0.14	0.16	0.18	0.16	0.22	0.26	0.29	0.32	0.28	0.34	0.38	0.40	0.43
	gplearn	0.11	0.16	0.20	0.22	0.25	0.27	0.33	0.37	0.40	0.44	0.38	0.44	0.48	0.51	0.54

Table 4. Robustness evaluation of SR models under different biased noise types and levels (1%–5%) for the expression-tree score. *Note:* Values are expression-tree distances.

the LLM does not dominate the total runtime. Moreover, all experiments were executed on a Google Colab Premium environment, suggesting that these runtimes are achievable with accessible computing resources.

Based on these results, we recommend the following practical considerations for integrating LLM into SR search of physical dynamics. First, adopt a fully informative prompt that includes clear variable definitions with their physical dimensions and a detailed experiment description. This configuration—corresponding to Prompt E in Table 2, which show promising results, as it both fitted the data well and find the GT equation exactly for all LLM and SR combinations. Second, consider the predictions of different SR as their computation is relatively cheap, and they seem to provide different results. Observing the common parts of the obtained equations from these can be useful in partial usage.

This study has several limitations that should be acknowledged. First, the quality and consistency of LLM-generated scores remain a challenge. The models were used in a zero-shot setting without fine-tuning, which likely limited their effectiveness. Although structured prompting was applied to encourage more reliable outputs, the non-deterministic nature of LLMs led to occasional invalid or syntactically incorrect suggestions, requiring prompt revisions and increasing training time. Second, the quality of LLM score was not manually or systematically evaluated by human experts. Without human verification, it is challenging to gauge whether performance gains were due to genuinely helpful scores in terms of physical realism or the score captures more technical properties, such as the equation's length. Third, our noise sensitivity analysis assumes a Gaussian noise with mean value equal to zero, which does not capture the full complexity of possible noise profiles, providing only an approximation to the noise robustness profile of the proposed method. Finally, the experiments were conducted on synthetically generated data with known ground truths. While this allows for controlled evaluation, it does not fully reflect the complexity and noise of real-world systems. For future work, we propose extending this investigation using more powerful, cloud-hosted LLMs, which may generate higher-quality and context-aware advice and thus better demonstrate the full potential of LLM performance, especially in more complex or higher-dimensional environments[104], and incorporating human-in-the-loop systems for expert validation. Moreover, as the LLM component alters the optimization landscape, future works can investigate and develop possible SR models that benefits from the LLM's feedback more comparable to current SR models.

Taken jointly, this study demonstrates that LLMs can be a straightforward way to introduce textual, physics-related context into SR via an auxiliary scoring term, potentially moving from fitting data toward generating more physically plausible hypotheses. When viewed in the larger context of automatic scientific discovery, where SR and LLMs have already taken a central role, this approach further deepens their connection to one another and the task. We emphasize that the LLM score is heuristic and does not guarantee physical validity; external validation remains necessary, especially for real-world datasets. Thus, more re- fined exploration of this relationship, taking into account the SR and LLM properties as well as the nature of the physical dynamics, should further reveal the optimal design of this method.

Data availability

The data and code are freely available in the following Github repository: https://github.com/ bilgesi/SR-LLM-I ntegration.

Received: 4 July 2025; Accepted: 5 January 2026
Published online: 13 January 2026

References

1. Merton, R. K. *The Sociology of Science: Theoretical and Empirical Investigations* (University of Chicago press, 1973).
2. Price, D. J. D. S. *Little Science, Big Science* (Columbia university press, 1963).
3. Redish, E. F. Using math in physics: overview. *Phys. Teacher* **59** (5), 314–318 (2021).
4. Yeargers, E. K., Herod, J. V. & Shonkweiler, R. W. *An Introduction to the Mathematics of Biology: with Computer Algebra Models* (Springer Science & Business Media, 2013).
5. Debreu, G. The mathematization of economic theory. *Am. Econ. Rev.* **81** (1), 1–7 (1991).
6. Maxwell, J. C. *Electricity and Magnetism* (Dover, 1954).
7. Einstein, A. The general theory of relativity. In *The Meaning of Relativity* 54–75 (Springer, 1922).
8. Gingras, Y. What did mathematics do to physics?. *Hist. Sci.* **39**(4), 383–416 (2001).
9. N. R. Council, D. on Engineering, P. Sciences, B. on Mathematical Sciences, T. Applications, C. on the Mathematical Sciences in, and 2025. *Fueling Innovation and Discovery: The Mathematical Sciences in the 21st Century* (National Academies Press, 2012).
10. Polya, G. *Mathematical Discovery, 1962* (Wiley, 1962).
11. Abrahamson, D., Gutierrez, J., Charoenying, T., Negrete, A. & Bumbacher, E. Fostering hooks and shifts: tutorial tactics for guided mathematical discovery. *Technol. Knowl. Learn.* **17**, 61–86 (2012).
12. Stam, J., Stokhof, M. & Van Lambalgen, M. Naturalising mathematics? A wittgensteinian perspective. *Philosophies* **7**, 85 (2022).
13. Holton, G., Chang, H. & Jurkowitz, E. How a scientific discovery is made: a case history. *Am. Sci.* **84**(4), 364–375 (1996).
14. T. Nickles. *Scientific Discovery, Logic, and Rationality* (Springer Science & Business Media, 2012).
15. Stevens, R. et al. Ai for science: Report on the department of energy (doe) town halls on artificial intelligence (ai) for science. Tech. rep. Argonne National Lab.(ANL) (2020).
16. Van Noorden, R. & Perkel, J. M. AI and science: what 1,600 researchers think. *Nature* **621** (7980), 672–675 (2023).
17. Krenn, M. et al. On scientific understanding with artificial intelligence. *Nat. Rev. Phys.* **4** (12), 761–769 (2022).
18. Grossmann, I. et al. AI and the transformation of social science research. *Science* **380**, 1108–1109 (2023).
19. Alyass, A., Turcotte, M. & Meyre, D. From big data analysis to personalized medicine for all: challenges and opportunities. *BMC Med. Genomics* **8**, 1–12 (2015).
20. Einav, L. & Levin, J. Economics in the age of big data. *Science* **346**(6210), 1243089 (2014).
21. Jiao, L. et al. AI meets physics: a comprehensive survey. *Artif. Intell. Rev.* **57**(9), 256 (2024).
22. Al-Baity, H. H. The artificial intelligence revolution in digital finance in Saudi Arabia: a comprehensive review and proposed framework. *Sustainability* **15**(18), 13725 (2023).
23. Pathak, P., Pal, P. R., Shrivastava, M. & Ora, P. Fifth revolution: applied AI & human intelligence with cyber physical systems. *Int. J. Eng. Adv. Technol.* **8** (3), 23–27 (2019).

24. Tjoa, E. & Guan, C. A survey on explainable artificial intelligence (xai): toward medical xai. *IEEE Trans. Neural Netw. Learn. systems* **32**(11), 4793–4813 (2020).
25. Petch, J., Di, S. & Nelson, W. Opening the black box: the promise and limitations of explainable machine learning in cardiology. *Can. J. Cardiol.* **38**(2), 204–213 (2022).
26. Azodi, C. B., Tang, J. & Shiu, S.-H. Opening the black box: interpretable machine learning for geneti- cists. *Trends Genet.* **36**(6), 442–455 (2020).
27. Pedreschi, D. et al. Meaningful explanations of black box AI decision systems. *Proc. AAAI Conf. Artif. Intell.* **33**, 9780–9784 (2019).
28. Holzinger, A., Langs, G., Denk, H., Zatloukal, K. & Muller, H. Causability and explainability of artificial intelligence in medicine. *Wiley Interdiscipl. Rev. Data Min. Knowl. Discov.* **9**, e1312 (2019).
29. Makke, N. & Chawla, S. Interpretable scientific discovery with symbolic regression: a review. *Artif. Intell. Rev.* **57**, 2 (2024).
30. Kronberger, G. et al. *Symbolic Regression* (CRC Press, 2024).
31. Udrescu, S.-M. & Tegmark, M. AI Feynman: a physics-inspired method for symbolic regression. *Sci. Adv.* **6**, eaay2631 (2020).
32. Schmidt, M. & Lipson, H. Distilling free-form natural laws from experimental data. *Science* **324**, 81–85 (2009).
33. Stalzer, M. & Ju, C. Automated rediscovery of the maxwell equations. *Appl. Sci.* **9**, 2899. https://doi.org/10.3390/app9142899 (2019).
34. Cranmer, M., Xu, R., Battaglia, P., Spergel, D. N. & Ho, S. Discovering symbolic models from deep learning with inductive biases. *Nat. Commun.* **11**, 5354. https://doi.org/10.1038/s41467-020-18020-0 (2020).
35. Udrescu, S.-M. & Tegmark, M. AI Feynman: a physics-inspired method for symbolic regression. *Science Adv.* **6**, eaay2631. https://doi.org/10.1126/sciadv.aay2631 (2020).
36. Brunton, S. L., Proctor, J. L. & Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl. Acad. Sci.* **113**, 3932–3937. https://doi.org/10.1073/pnas.1517384113 (2016).
37. Keren, L. S., Liberzon, A. & Lazebnik, T. A computational framework for physics-informed symbolic regression with straightforward integration of domain knowledge. *Sci. Rep.* **13**, 1249 (2023).
38. Kubalık, J., Derner, E. & Babuska, R. Multi-objective symbolic regression for physics-aware dynamic modeling. *Expert Syst. Appl.* **182**, 115210 (2021).
39. Kubalık, J., Derner, E. & Babuska, R. Toward physically plausible data-driven models: a novel neural network approach to symbolic regression. *IEEE Access* **11**, 61481–61501 (2023).
40. Platt, M. & Platt,D. Effectiveness of generative artificial intelligence for scientific content analysis. In *2023 IEEE 17th International Conference on Application of Information and Communication Technologies (AICT)* 1–4 (IEEE. 2023).
41. Icke, I. & Bongard, J. Modeling hierarchy using symbolic regression. In *2013 IEEE Congress on Evolutionary Computation* 2980–2987 (2013). https://doi.org/10.1109/cec.2013.6557932.
42. Kammerer, L. et al. Symbolic regression by exhaustive search: reducing the search space using syntactical constraints and efficient semantic structure deduplication. In *Genetic and Evolutionary Computation* 79–99 (2020). https://doi.org/10.1007/978-3-030-39958-0_5.
43. Valipour, M., You, B., Panju, M.& Ghodsi, A. Symbolicgpt: a generative transformer model for symbolic regression. arXiv preprint arXiv:2106.14131 (2021).
44. Makke, N. & Chawla, S. Interpretable scientific discovery with symbolic regression: a review. *Artif. Intell. Rev.* **57**, 1. https://doi.org/10.1007/s10462-023-10622-0 (2024).
45. Chen, C., Luo, C. & Jiang, Z. Block building programming for symbolic regression. *Neurocomputing* **275**, 1973–1980. https://doi.org/10.1016/j.neucom.2017.10.047 (2018).
46. Jin, Y., Fu, W., Kang, J., Guo, J. & Guo, J. Bayesian symbolic regression. https://doi.org/10.48550/arxiv.1910.08892 (2019).
47. Tohme, T., Liu, D. & Youcef-Toumi, K. Gsr: a generalized symbolic regression approach (2022). https://doi.org/10.48550/arxiv.2205.15569.
48. Keren, L. S., Liberzon, A. & Lazebnik, T. A computational framework for physics-informed symbolic regression with straightforward integration of domain knowledge. *Sci. Rep.* **13**, 12. https://doi.org/10.1038/s41598-023-28328-2 (2023).
49. Riolo,R. *Genetic Programming Theory and Practice X* (Springer, 2013).
50. Heule, M. J. & Kullmann, O. The science of brute force. *Commun. ACM* **60**, 70–79 (2017).
51. Brunton, S. L., Proctor, J. L. & Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl. Acad. Sci.* **113**, 3932–3937 (2016).
52. Kaptanoglu, A. A. et al. PySINDy: a comprehensive Python package for robust sparse system identification. In arXiv preprint arXiv:2111.08481 (2021).
53. La Cava, W. et al. Contemporary symbolic regression methods and their relative performance. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS 2021) Datasets and Benchmarks Track* (2021).
54. Petersen, B. K. et al. Deep symbolic regression: recovering mathematical expressions from data via risk-seeking policy gradients (2019). https://doi.org/10.48550/arxiv.1912.04871.
55. Orzechowski, P., Cava, W. L. & Moore, J. H. Where are we now?" In *Proceedings of the Genetic and Evolutionary Computation Conference* (2018). https://doi.org/10.1145/3205455.3205539.
56. Virgolin, M., Alderliesten, T., Witteveen, C. & Bosman, P. A. N. Improving model-based genetic programming for symbolic regression of small expressions. *Evol. Comput.* **29**, 211–237. https://doi.org/10.1162/evco_a_00278 (2021).
57. Stephens, T. et al. Genetic Programming in Python, with a scikit-learn inspired API: gplearn". In Documentation at https://gplearnreadthedocs.io/en/stable/intro.html (2016).
58. Tohme, T., Liu, D. & Youcef-Toumi,K. GSR: a generalized symbolic regression approach. CoRR abs/2205.15569 (2022). https://doi.org/10.48550/ARXIV.2205.15569.
59. Landajuela, M. et al. A unified framework for deep symbolic regression. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)* (2022). http://papers.nips.cc/paper_files/paper/2022/hash/dbca58f35bddc6e4003b2dd80e42f838-Abstract-Conference.html.
60. Jiang, N., Nasim, M. & Xue.Y. Vertical symbolic regression via deep policy gradient. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI-24)* 5891–5899 (2024). https://doi.org/10.24963/ijcai.2024/651.
61. Neumann, P., Cao, L., Russo, D., Vassiliadis, V. S. & Lapkin, A. A. A new formulation for symbolic regression to identify physico-chemical laws from experimental data. *Chem. Eng. J.* **387**, 123412. https://doi.org/10.1016/j.cej.2019.123412 (2020).
62. Kim, S. et al. Integration of neural network-based symbolic regression in deep learning for scientific discovery. 10.48550/arxiv.1912.04825 (2019).
63. Jiang, N. & Xue, Y. Racing control variable genetic programming for symbolic regression. *Proc. AAAI Conf. Artif. Intel.* **38**, 12901–12909. https://doi.org/10.1609/aaai.v38i11.29187 (2024).
64. Kubalık, J., Derner, E. & Babuska, R. Multi-objective symbolic regression for physics-aware dynamic modeling. *Expert Syst. Appl.* **182**, 115210. https://doi.org/10.1016/j.eswa.2021.115210 (2021).
65. Morales-Alvarado, M., Conde, D., Bendavid, J., Sanz, V. & Ubiali, M. Symbolic regression for precision LHC physics. Uses PySR as the symbolic-regression engine. *arXiv* 2412.07839 (2024). https://arxiv.org/abs/2412.07839.
66. Karniadakis, G. E. et al. Physics-informed ma- chine learning. *Nat. Rev. Phys.* **3**, 422–440. https://doi.org/10.1038/s42254-021-00314-5 (2021).

67. Raissi, M., Perdikaris, P. & Karniadakis, G. E. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **378**, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045 (2019).
68. Meng, C., Griesemer, S., Cao, D., Seo, S. & Liu, Y. When physics meets machine learning: a survey of physics-informed machine learning. *Mach. Learn. Comput. Sci. Eng.* **1**, 20. https://doi.org/10.1016/j.mlcse.2024.100020 (2025).
69. Makke, N. & Chawla, S. Data-driven discovery of tsallis-like distribution using symbolic regression in high-energy physics. *PNAS Nexus* **3**, 11. https://doi.org/10.1093/pnasnexus/pgae467 (2024).
70. Tenachi, W., Ibata, R. & Diakogiannis, F. I. Deep symbolic regression for physics guided by units con- straints: toward the automated discovery of physical laws. *Astrophys. J.* **959**, 99. https://doi.org/10.3847/1538-4357/ad014c (2023).
71. Udrescu, S. & Tegmark, M. Ai feynman: a physics-inspired method for symbolic regression. *Sci. Adv.* **6**, 25. https://doi.org/10.1126/sciadv.aay2631 (2020).
72. Chen, Z., Liu, Y. & Sun, H. Physics-informed learning of governing equations from scarce data. *Nat. Commun.* **12**, 6136 (2021).
73. Martius, G. & Lampert,C. H. Extrapolation and learning equations. arXiv preprint arXiv:1610.02995 (2016).
74. Golden, M. Scalable sparse regression for model discovery: the fast lane to insight arXiv preprint arXiv:2405.09579 (2024).
75. Hou, Y. et al. Large language models are zero- shot rankers for recommender systems. In *European Conference on Information Retrieval* 364–381 (Springer, 2024).
76. Wu, L. et al. A survey on large language models for recommendation. arXiv preprint arXiv:2305.19860 (2023).
77. Zhao, Z. et al. Recommender systems in the era of large language models (llms). *IEEE Trans. Knowl. Data Engineering* **2024**, 256 (2024).
78. Guo, M., Hwa, R. & Kovashka,A. Decoding symbolism in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2023). https://doi.org/10.18653/v1/2023.acl-long.186.
79. Espinosa, L. et al. Open-source solution for evaluation and benchmarking of large language models for public health (2025). 10.1101/ 2025.03.20.25324040.
80. Hsu, J. C. et al. Applications of advanced natural language processing for clinical pharmacology. *Clin. Pharmacol. Therapeut.* **115**, 786–794. https://doi.org/10.1002/cpt.3161 (2024).
81. Kim, S. H. et al. Large language models meet collaborative filtering: an efficient all-round LLM-based recommender system. arXiv preprint arXiv:2404.11343 (2024).
82. Liu, Q., Chen, N., Sakai, T. & Wu, X.-M. Once: boosting content-based recommendation with both open- and closed-source large language models . In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining* 452–461 (2024).
83. Harte, J. et al. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems* 1096–1102 (2003).
84. Li, P., de Rijke, M., Xue, H., Ao, S., Song, Y. & Salim, F. D. Large Language models for next point-of- interest recommendation. arXiv preprint arXiv:2404.17591 (2024).
85. Wu, J. C. et al. CoRAL: collaborative retrieval- augmented large language models improve long-tail recommendation. In arXiv preprint arXiv:2403.06447 (2024).
86. Sanner, S., Balog, K., Radlinski, F., Wedin, B. & Dixon, L. Large language models are competitive near cold-start recommenders for language-and item-based preferences. In *Proceedings of the 17th ACM Conference on Recommender Systems* 890–896 (2023).
87. Kasneci, E. et al. ChatGPT for good? On opportunities and challenges of large language models for education. *Learn. Individ. Differ.* **103**, 102274 (2023).
88. Thirunavukarasu, A. J. et al. Large language models in medicine. *Nat. Med.* **29**, 1930–1940 (2023).
89. M. U. Hadi, R. Qureshi, A. Shah, M. Irfan, A. Zafar, M. B. Shaikh, N. Akhtar, J. Wu, S. Mirjalili, et al. "A survey on large language models: Applications, challenges, limitations, and practical usage". *Authorea Preprints* (2023).
90. Demszky, D. et al. Using large language models in psychology. *Nat. Rev. Psychol.* **2**, 688–701 (2023).
91. Wei, C., Xie, S. M. & Ma, T. Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning. *Adv. Neural. Inf. Process. Syst.* **34**, 16158–16170 (2021).
92. Merler, M., Haitsiukevich, K., Dainese, N. & Marttinen, P. In-context symbolic regression: leveraging large language models for function discovery. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*. Association for Computational Linguistics 589–606 (2024). https://doi.org/10.18653/v1/2024.acl.
93. Sharlin, S. & Josephson, T. R. In context learning and reasoning for symbolic regression with large language models. In: arXiv preprint arXiv:2410.17448 (2024).
94. Sharlin, S. & Josephson, T. R. In *Context Learning and Reasoning for Symbolic Regression with Large Language Models* (2025). arXiv: 2410.17448 [cs.CL].
95. Li, Y. et al. *MLLM-SR: Conversational Symbolic Regression base Multi-Modal Large Language Models*. arXiv: 2406.05410 [cs.AI] (2024).
96. Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (MIT Press, 1992).
97. Schmidt, M. & Lipson, H. Distilling free-form natural laws from experimental data. *Science* **324**, 81–85. https://doi.org/10.1126/science.1165893 (2009).
98. Goldstein, H., Poole, C. P. & Safko,J. L. *Classical Mechanics* (Addison-Wesley, 2001).
99. Jackson, J. D. *Classical Electrodynamics* (Wiley, 1999).
100. Keijzer, M. Improving symbolic regression with interval arithmetic and linear scaling. *Eur. Conf. Genet. Program.* **2610**, 70–82 (2003).
101. Uy, N. Q., Hoai, N. X., O'Neill, M., McKay, R. & Galvan-Lopez, E. Semantically-based crossover in genetic programming: application to real-valued symbolic regression. *Genet. Program. Evol. Mach.* **12**, 91–119 (2011).
102. Vladislavleva, E. J., Smits, G. F. & den Hertog, D. Order of nonlinearity as a complexity measure for models generated by symbolic regression via Pareto genetic programming. *IEEE Trans. Evol. Comput.* **13**, 333–349 (2009).
103. Reinbold, P. A., Beloin, D. & Haghverdi, L. A robust symbolic regression benchmark with noise, extrapolation, and overfitting controls. *Genet. Program Evolvable Mach.* **22**, 205–239. https://doi.org/10.1007/s10710-020-09384-2 (2021).
104. Shojaee, N. et al. LLM-SR: guiding symbolic regression with large language models. arXiv preprint arXiv:2502.XXXXX (2025).
105. Romera-Paredes, B. et al. FunSearch: bridging program synthesis and discovery with LLMs. *Nature* (2023).
106. Merler, M. et al. In-context symbolic regression with LLMs. In arXiv preprint arXiv:2401.XXXXX (2024).
107. Rae, J. et al. Scaling language models: methods, analysis & interpretation. In arXiv preprint arXiv:2109.04486 (2021).
108. OpenAI. GPT-4 Technical Report . In arXiv preprint arXiv:2303.08774 (2023).
109. Chen, Y. et al. FrugalGPT: how to use a large language model cheaply and efficiently. In arXiv preprint arXiv:2305.05176 (2023).
110. Yuan, S., Feng, L. & Liu, T. Early stopping against label noise without validation data. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)* (2024).
111. Zhu, K. et al. PromptBench: towards evaluating the robustness of large language models on adversarial prompts. In arXiv preprint arXiv:2306.04528 (2023).
112. Cao, B., Cai, D., Zhang, Z., Zou, Y. & Lam, W. On the worst prompt performance of large language models. In *Advances in Neural Information Processing Systems (NeurIPS 2024)* (2024).
113. Sclar, M., Choi, Y., Tsvetkov, Y. & Suhr, A. Quantifying language models' sensitivity to spurious features in prompt design, or: how i learned to start worrying about prompt formatting. In arXiv preprint arXiv:2310.11324 (2023).

114. Raj, H., Gupta, V., Rosati, D. & Majumdar, S. Semantic consistency for assuring reliability of large language models. In: arXiv preprint arXiv:2308.09138 (2023).
115. Zhuo, J. et al. ProSA: assessing and understanding the prompt sensitivity of LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024* 1950–1976 (2024).
116. Matsubara, Y., Chiba, N., Igarashi, R. & Ushiku, Y. Rethinking symbolic regression datasets and benchmarks for scientific discovery. In *Journal of Data-centric Machine Learning Research* (2024). https://openreview.net/forum?id=qrUdrXsiXX.
117. Bertschinger, A, Davis, Q. T, Bagrow, J. P.& Bongard, J. C. The metric is the message: benchmarking challenges for neural symbolic regression. In *Machine Learning and Knowledge Discovery in Databases: Research Track (ECML PKDD 2023, Part IV). Vol. 14172. Lecture Notes in Computer Science* (Springer, 2023) https://doi.org/10.1007/978-3-031-43421-1_10.
118. dos Reis, L. G. A., Caminha, V. L. P. S. & Penna, T. J. P. Benchmarking symbolic regression constant optimization schemes. In arXiv preprint arXiv:2412.02126 (2024).
119. Ma, R., Luijkx, J., Ajanovic, Z. & Kober, J. *ExploRLLM: Guiding Exploration in Reinforcement Learning with Large Language Models*. arXiv: 2403.09583 [cs.RO] (2024).
120. Zhou, Z., Hu, B., Zhao, C., Zhang, P. & Liu, B. *Large Language Model as a Policy Teacher for Training Reinforcement Learning Agents*. arXiv: 2311.13373 [cs.AI] (2024).
121. Castillo, E., Mınguez, R. & Castillo, C. Sensitivity analysis in optimization and reliability problems. *Reliab. Eng. Syst. Saf.* **93**, 1788–1800 (2008).
122. Liu, X., Zhang, Y. & Wang, J. Pre-train prompt tuning: towards effective llm guidance. In arXiv preprint arXiv:2302.00000 (2023).
123. Schick, T. & Schutze, H. Exploiting cloze questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)* 255–269 (2021).
124. Wang, X. et al. Self-consistency improves chain of thought reasoning in language models. In arXiv preprint arXiv:2203.11171 (2022).

Author contributions
Original Draft, Writing—Review & Editing. Wenxiong Xie: Software, Formal analysis, Writing—Original Draft. Teddy Lazebnik: Conceptualization, Methodology, Formal analysis, Validation, Writing—Original Draft, Writing—Review & Editing, Visualization, Project administration.

Funding
Open access funding provided by Jönköping University.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-026-35327-6.

Correspondence and requests for materials should be addressed to T.L.