# MINT: SHARED PDE IDENTIFICATION WITH NEURAL FIELDS AND GENETIC PROGRAMMING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Identifying a partial differential equation (PDE) from sparse measurements is difficult because different equations can fit the same observed solution. We propose MINT (Multi-trajectory Identification with Neural fields and Trees), which fits independent neural fields and uses genetic programming (GP) to construct a single equation shared across multiple trajectories. MINT searches for terms and coefficients simultaneously and retains the selected structure while refining the fields and coefficients. We also develop MINT-M, which uses additional family-specific motifs to guide initialization. Across six benchmarks with 30 runs each, MINT recovers the terms of 2D and 3D advection in 86.7% and 43.3% of noise-free runs, respectively, whereas two GP-based baselines recover neither. In the same setting, MINT-M recovers Burgers and Fisher–KPP terms in 56.7% and 63.3% of runs, respectively, compared with no recoveries for MINT. Both methods also recover 2D advection more often than the GP baselines at 1% and 5% noise. With 250 observations per trajectory, both improve field accuracy over the common data-only reconstruction on 2D advection, Burgers, and Fisher at both noise levels. Missing-library and noise tests further show that accurate fields and low residuals alone do not establish recovery of the governing law.

## 1 INTRODUCTION

Partial differential equations (PDEs) are a fundamental language for describing how physical quantities evolve across space and time, underlying models of fluid flow, transport, diffusion, and many other dynamical systems. However, in many practical settings, the governing equation is not fully known (Rudy et al., 2017) due to the fact that relevant physical mechanisms may be only partially understood, first-principles derivations may be difficult, or existing models may omit interactions that become important under new operating conditions. Meanwhile, modern experiments and simulations increasingly provide measurements of the resulting system dynamics. These observations motivate PDE discovery, which aims to recover explicit governing equations directly from data.

Recent advances in machine learning have shown considerable potential for discovering PDEs directly from observational data. Early approaches based on inverse physics-informed neural networks (PINNs) infer unknown physical parameters by embedding a prescribed governing equation into neural-network training, but generally require the equation structure itself to be specified in advance (Raissi et al., 2019). More recent methods relax this requirement by allowing the equation structure to be identified from data. For instance, sparse-regression approaches (Rudy et al., 2017; Messenger & Bortz, 2021; Both et al., 2021; Chen et al., 2021) construct a predefined library of candidate functions and derivatives and select a sparse subset to form the governing equation. While effective in many settings, these methods still require the relevant physical interactions to be represented in the candidate library. This requirement can be restrictive when the underlying mechanisms are only partially known, since the nonlinear interactions or derivative combinations that constitute the governing equation may themselves be unknown. A more flexible alternative is therefore to construct candidate equations rather than select them from a fixed library. Genetic programming (GP) provides such a mechanism by recursively combining field variables, derivatives, and mathematical operators into symbolic expressions, allowing composite terms to emerge during the search rather than being enumerated beforehand (Schmidt & Lipson, 2009; Xu et al., 2020). Motivated by this flexibility, we investigate GP-based PDE discovery in a setting where the governing equation structure is not assumed to be known in advance.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

However, the increased flexibility of GP introduces a new challenge: a sufficiently expressive symbolic search can produce many low-residual equations that fit the observed solution but do not represent the underlying governing dynamics. This makes it difficult to distinguish the true governing structure from solution-specific or spurious relations when only a single solution is observed. In many physical systems, however, the same governing PDE can generate different space–time solutions under different initial conditions, boundary conditions, or external excitations. We refer to these distinct solutions as trajectories. Although their observed fields may differ substantially, the underlying governing equation remains unchanged. Relations that arise only from the particular form of one solution may therefore fail to hold for other trajectories, while the governing equation should remain valid across all of them. Based on this observation, we propose MINT, a multi-trajectory PDE discovery framework that constrains flexible symbolic search using multiple solutions governed by the same PDE. Our main contributions are summarized as follows:

- We propose MINT, a multi-trajectory PDE discovery framework that combines neural field reconstruction with GP-based symbolic search. By searching for a single governing equation shared across multiple trajectories generated by the same PDE, MINT leverages cross-solution consistency to constrain flexible symbolic discovery.
- We develop a two-stage discovery procedure that separates discrete equation-structure search from continuous refinement. Neural fields are first fitted to approximate the solution trajectories and then fixed during GP search. After the PDE structure is identified, the symbolic structure is retained while the equation coefficients and neural fields are jointly refined.
- We provide a systematic evaluation across six PDE benchmarks under variations in structural motifs, missing candidate terms, and measurement noise. Beyond demonstrating the benefits of multi-trajectory symbolic discovery, our results expose its limitations: accurate field reconstruction and low residuals do not necessarily imply correct recovery of the governing equation.

## 2 Related Work

**Physics-informed neural network for inverse problems.** Early learning-based approaches often formulate PDE identification as an inverse problem in which the governing equation is known and only unknown physical parameters need to be estimated. Physics-informed neural networks (PINNs), for example, embed a prescribed PDE into neural-network training and infer its unknown coefficients from observational data (Raissi et al., 2019). This formulation is effective when reliable prior knowledge of the governing form is available, but it cannot directly discover an unknown equation structure.

**Sparse PDE discovery.** Sparse identification methods relax this assumption by allowing the active equation terms to be selected from data. SINDy and PDE-FIND construct a predefined library of candidate functions and derivatives and identify a sparse subset that explains the observed dynamics (Brunton et al., 2016; Rudy et al., 2017; Schaeffer, 2017). Subsequent approaches improve robustness to sparse or noisy observations. WSINDy reduces sensitivity to derivative estimation through weak formulations (Messenger & Bortz, 2021), while DeepMoD and PINN-SR combine neural field reconstruction with sparse term selection (Both et al., 2021; Chen et al., 2021). These methods enable structure discovery beyond coefficient estimation, but the discoverable PDE is still restricted by the candidate terms specified in advance.

**Symbolic PDE discovery.** Symbolic regression further relaxes this restriction by constructing mathematical expressions instead of selecting terms only from a predefined library (Schmidt & Lipson, 2009; Petersen et al., 2021; Biggio et al., 2021; Cranmer et al., 2020). Genetic programming is particularly suitable for this setting because field variables, derivatives, and operators can be recursively combined to generate new candidate structures. DLGA-PDE applies this idea to PDE discovery by coupling neural field approximation with genetic search over derivative genes and their products, allowing composite terms to emerge during evolution (Xu et al., 2020). However, the increased flexibility also enlarges the space of low-residual expressions that may fit an observed solution, making it harder to distinguish the governing PDE from solution-specific alternatives. MINT addresses this issue by using multiple solutions governed by the same PDE to constrain symbolic search and by separating discrete structure discovery from subsequent continuous refinement.

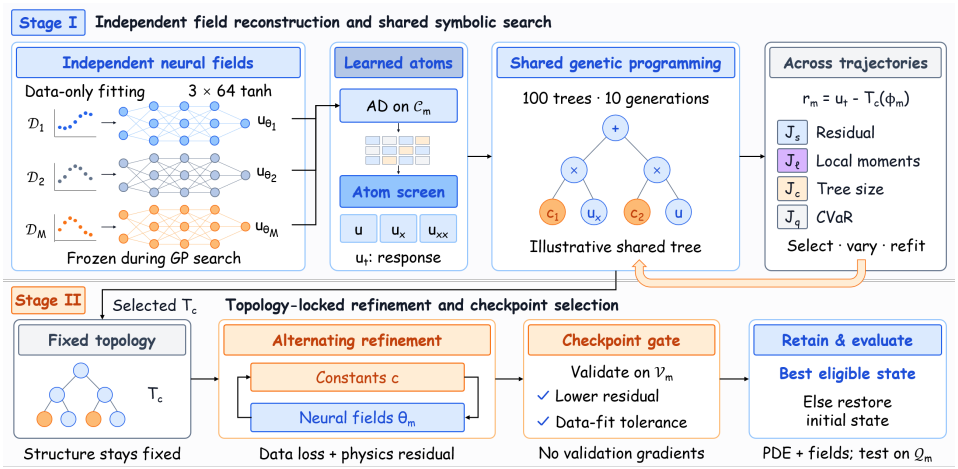


Figure 1: The overall framework of the proposed MINT.

## 3 Problem Formulation

We consider $M$ solution trajectories generated by the same autonomous scalar PDE,

$$\partial_t u^{(m)} = \mathcal{F}\left(u^{(m)}, \nabla u^{(m)}, \nabla^2 u^{(m)}, \dots\right), \qquad m = 1, \dots, M, \tag{1}$$

where the trajectories may arise from different initial or boundary conditions but share the same governing equation. For each trajectory, we observe only sparse field measurements $\mathcal{D}_m = \{(\mathbf{z}_{mi}, y_{mi})\}_{i=1}^{N_o}$, where $\mathbf{z}_{mi} = (\mathbf{x}_{mi}, t_{mi})$ denotes a space–time coordinate and $y_{mi} = u^{(m)}(\mathbf{z}_{mi}) + \epsilon_{mi}$. No spatial or temporal derivatives are directly observed.

Given $\{\mathcal{D}_m\}_{m=1}^{M}$, our goal is to identify both the symbolic structure and coefficients of the shared governing PDE. This setting differs from inverse PINNs, which assume a prescribed equation structure and estimate only its unknown parameters (Raissi et al., 2019), and from sparse PDE discovery methods, which select active terms from a predefined candidate library. Here, the equation structure is constructed through symbolic search within a specified grammar, while neural fields are used to approximate the observed solutions and provide the derivatives required for evaluating candidate equations.

## 4 The MINT Framework

MINT is designed to identify a shared governing PDE from sparse observations of multiple solution trajectories while reducing the ambiguity introduced by flexible symbolic search. The framework represents each trajectory with an independent neural field $u_{\theta_m}$ and represents the shared governing law with a single GP expression tree $T_c$. This design separates trajectory-specific solution reconstruction from equation discovery: the neural fields capture the different observed solutions, whereas the shared tree enforces a common PDE structure and coefficients across trajectories.

In general, MINT consists of two stages with complementary roles (Figure 1). In Stage I, the neural fields are first fitted to the observations and then fixed while GP searches for the shared symbolic structure. Fixing the fields ensures that all candidate equations are evaluated using the same field and derivative estimates, while cross-trajectory evaluation discourages structures that explain only individual solutions. In Stage II, the selected tree structure is fixed and only the continuous quantities, including the neural-field parameters and equation coefficients, are refined. This separation allows MINT to first perform stable discrete structure discovery and then use the identified PDE to further improve the continuous fit without reopening the symbolic search.

### 4.1 Stage I: Symbolic Discovery of a Shared PDE Tree

Stage I aims to identify a shared PDE structure while keeping the continuous representation of each trajectory fixed. It consists of two steps. First, an independent neural field is fitted to each trajectory to obtain a differentiable approximation of the observed solution and its derivatives. The neural fields are then frozen. Second, GP searches for a single symbolic expression shared across all trajectories. Fixing the neural fields ensures that all candidate equations are evaluated using the same field and derivative estimates throughout the symbolic search.

**Neural field construction.** For each trajectory $m$, we approximate the corresponding solution using an independent neural field $u_{\theta_m}$. The neural fields are fitted using only the sparse observations in $\mathcal{D}_m$ by minimizing

$$\mathcal{L}_{\text{data}} = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{N_o}\sum_{(\mathbf{z},y)\in\mathcal{D}_m}\left(u_{\theta_m}(\mathbf{z})-y\right)^2. \tag{2}$$

After training, automatic differentiation is used to evaluate the temporal derivative $\partial_t u_{\theta_m}$ and the candidate spatial quantities required for equation discovery. We collect the candidate inputs at coordinate $\mathbf{z}$ into $\phi_m(\mathbf{z})$, which may include $u_{\theta_m}(\mathbf{z})$, $\nabla u_{\theta_m}(\mathbf{z})$, $\nabla^2 u_{\theta_m}(\mathbf{z})$, and other admissible derivatives defined by the symbolic grammar.

**Symbolic representation of the PDE.** The neural-field parameters $\theta_m$ are then fixed throughout the GP search. For a candidate expression tree $T_c$, where $c$ denotes its continuous constants, the residual for trajectory $m$ is

$$r_m(\mathbf{z};T_c,\theta_m) = \partial_t u_{\theta_m}(\mathbf{z}) - T_c(\phi_m(\mathbf{z})). \tag{3}$$

All candidate equations are therefore evaluated using the same neural-field approximations and derivative estimates.

GP searches for a single expression tree $T_c$ shared across all $M$ trajectories. Before evolution, candidate atoms are screened to reduce the symbolic search space. The shared terminal set is formed from atoms selected by an anchored sparse fit together with atoms exhibiting strong absolute correlations with the temporal derivative across trajectories. The temporal derivative is used only as the response and is excluded from the terminal set. This screening step determines the primitive quantities available to GP but does not determine the final governing terms. Given the screened terminal set, GP constructs candidate expressions using the operator set $\{+,-,\times,\mathrm{pdiv},(\cdot)^2,\mathrm{neg}\}$, where $\mathrm{pdiv}$ denotes protected division and $\mathrm{neg}$ denotes unary negation. The initial population contains randomly generated trees, individual atoms, scaled atoms, and pairwise products. For each candidate tree structure, the constants $c$ are optimized subject to bounded constraints and are shared across all trajectories.

**Cross-trajectory fitness.** A candidate equation should explain all trajectories consistently rather than achieve a small residual on only a subset of them. The fitness therefore evaluates four complementary properties: global residual accuracy, local residual consistency, expression complexity, and performance on poorly fitted trajectories.

Let $\mathcal{C}_m$ denote the discovery coordinates for trajectory $m$, with $N_c = |\mathcal{C}_m|$, and let $r_m \in \mathbb{R}^{N_c}$ denote the residual vector obtained from Eq. equation 3. To reduce scale differences across trajectories, we define

$$J_s = \frac{1}{M}\sum_{m=1}^{M} s_m^2, \tag{4}$$

where $\widetilde{r}_m = r_m/\max\{\gamma_m, 10^{-12}\}$, $s_m^2 = \frac{1}{N_c}\|\widetilde{r}_m\|_2^2$, and $\gamma_m$ is the largest root-mean-square magnitude among the candidate feature columns evaluated on $\mathcal{C}_m$. Here, $J_s$ measures the mean normalized PDE residual across trajectories.

A low global residual may still hide systematic errors within local regions. We therefore introduce a local residual term together with an upper-tail trajectory loss:

$$J_\ell = \frac{1}{MK}\sum_{m=1}^{M}\sum_{k=1}^{K}\left(\frac{w_{mk}^{\mathsf{T}}\widetilde{r}_m}{\sqrt{N_c}}\right)^2, \qquad J_q = \mathrm{CVaR}_\alpha\left(\{s_m^2\}_{m=1}^{M}\right). \tag{5}$$

Here, $w_{mk} \in \mathbb{R}^{N_c}$ is the normalized weight vector associated with the $k$-th local window of trajectory $m$. The windowed moments measure signed residuals over local regions of the solution domain and help detect systematic mismatch that may be obscured by the global average. The upper-tail loss $J_q$ prevents the search from favoring equations that fit most trajectories well while leaving a small number poorly explained. In practice, it is computed as the mean of the largest $\lceil \alpha M \rceil$ values among $\{s_m^2\}_{m=1}^M$, where $\alpha$ denotes the upper-tail fraction.

Tree complexity is measured by $J_c = |T_c|$, where $|T_c|$ denotes the number of nodes in the expression tree. The overall GP fitness is then

$$J(T_c) = \frac{J_s}{a_s} + \frac{J_\ell}{a_\ell} + 0.05\frac{J_c}{a_c} + \frac{J_q}{a_q}, \tag{6}$$

where $a_s$, $a_\ell$, $a_c$, and $a_q$ are positive component scales that remain fixed throughout evolution. The terms $J_s$ and $J_\ell$ measure global and local residual agreement, respectively, $J_q$ promotes consistency across trajectories, and $J_c$ discourages unnecessarily complex expressions.

With the neural fields, symbolic representation, and cross-trajectory fitness defined above, GP evolves the population of expression trees through iterative selection and variation. The best-performing tree at the end of evolution is taken as the discovered PDE structure, with its associated shared constants providing the corresponding coefficient estimates. Stage II then fixes this symbolic structure and further refines the continuous parameters, including the equation constants and neural-field parameters.

### 4.2 Stage II: Refinement with a Fixed Tree

Stage II compiles the selected tree into a differentiable program while keeping its symbolic structure fixed. The neural-field parameters and shared constants are then updated alternately. For fixed constants $c$, the field parameters are optimized by minimizing

$$\mathcal{L}_{\text{ref}} = \sum_{m=1}^{M} \left[ \mathbb{E}_{(\mathbf{z},y)\in\mathcal{D}_m} \left(u_{\theta_m}(\mathbf{z}) - y\right)^2 + \lambda_p \mathbb{E}_{\mathbf{z}\in\mathcal{C}_m} r_m(\mathbf{z}; T_c, \theta_m)^2 \right], \tag{7}$$

where each expectation denotes an empirical average over the indicated set. The weight $\lambda_p$ controls the influence of the PDE residual relative to the observation error.

At each constant-update step, MINT recomputes the field values and derivatives from the current neural fields, treats them as fixed, and updates only $c$ by minimizing $\sum_{m=1}^{M} \mathbb{E}_{\mathbf{z}\in\mathcal{C}_m} r_m(\mathbf{z}; T_c, \theta_m)^2$. A field update is performed at every refinement step using freshly computed derivatives while holding $c$ fixed. The neural-field parameters and equation constants are thus optimized through separate updates, while the tree structure remains unchanged.

**Checkpoint selection.** Refinement can reduce the PDE residual by moving the neural fields away from the observations. To limit this behavior, MINT selects the returned state using both observation fidelity and residuals evaluated at held-out validation coordinates. At checkpoint $k$, let

$$D_k = \mathcal{L}_{\text{data}}^{(k)}, \qquad P_k = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\mathbf{z}\in\mathcal{V}_m} r_m(\mathbf{z}; T_c, \theta_m)^2. \tag{8}$$

Both losses are averaged across trajectories. The validation loss $P_k$ uses the raw PDE residual rather than the normalized GP fitness used in Stage I.

Let $D_0$ and $P_0$ denote the corresponding losses before refinement. A checkpoint is eligible for retention if

$$D_k \le \tau_d D_0 + \varepsilon, \qquad \frac{P_0 - P_k}{\max(P_0, \varepsilon)} \ge \tau_p, \tag{9}$$

where the first condition bounds deterioration in observation fit, while the second requires a minimum relative reduction in validation residual. Here, $\tau_d$ and $\tau_p$ are the corresponding thresholds, and $\varepsilon = 10^{-12}$ provides numerical stability.

MINT evaluates these conditions at fixed intervals and at the final refinement step. Among all eligible checkpoints, it retains the one with the smallest validation residual $P_k$. If no checkpoint

| Prior information | Methods | Structure search | Unknowns |
|---|---|---|---|
| Atoms and grammar | MINT, Plain GP, SP-Prune | Grammar-wide | Structure, constants |
| Grammar + motifs | MINT-M | Grammar-wide | Structure, constants |
| Complete library | DLGA-style mask, PDE-FIND-style, weak regression, DeepMoD-style, linear Hybrid | Library-limited | Active terms, coefficients |
| Exact active terms | Inverse PINN (standard, staged) | Fixed support | Non-anchor coefficients |

Table 1: Structural priors and discovery tasks. A complete library contains the true terms but does not specify their activity. Motifs guide initialization without restricting the grammar.

satisfies the retention criteria, the pre-refinement neural fields, equation constants, and optimizer states are restored. These checks determine only the returned state; they do not interrupt optimization or reject individual refinement steps.

# 5 Experimental Setup

**Benchmarks.** We use six PDEs with analytic solutions: heat and advection in two and three dimensions (H2/H3 and A2/A3), Burgers, and Fisher–KPP. Each has three trajectories, except H3, which has four. For the selected separated heat modes, the fourth trajectory distinguishes a possible reaction term from a linear combination of the three directional diffusion terms. The main experiment uses noise-free observations, with analytic solutions providing the evaluation references. The nonlinear benchmarks contain only traveling waves. Appendix C gives the governing equations, trajectory parameters, and domains.

**Baselines and prior information.** Table 1 groups the eleven tested implementations by their structural inputs. MINT uses no family-specific motifs, whereas MINT-M adds them to the initial population. Inverse PINNs receive the true active terms, so their support is marked *Given*. The linear Hybrid is not applicable to either nonlinear PDE, and the weak-regression implementation is not applicable to the rank-deficient Fisher wave library; these cases are marked N/A. The baselines are local adaptations to the shared observation setting, and our comparisons concern these implementations, detailed in Appendix E.

**Training and search settings.** Each trajectory has 500 observations and 2,000 coordinates in each discovery, validation, and residual-test set. Networks use three 64-unit $\tanh$ layers. MINT uses 4,500 data-only updates, a population of 100 evolved for 10 GP generations, and up to 500 refinement steps. Plain GP, SP-Prune, and DLGA-style share the evolutionary budget. Standard inverse PINN uses 5,000 updates with data and physics losses; staged inverse PINN and DeepMoD-style use 4,500 data-only and 500 joint updates. Five methods share the same frozen field reconstruction within each PDE and run. Appendix B describes data generation and provides details for reproducing the experiments.

**Metrics and statistical comparisons.** We use 30 runs per method–PDE combination, totaling 1,890 completed runs and 90 predeclared N/A entries. Exact support recovery requires matching the reference terms on the right-hand side after expansion and removal of coefficients with magnitude at most $10^{-4}$. Non-polynomial outputs count as recovery failures. Field error is the relative $L_2$ error against the analytic solution; it measures reconstruction. Field-error tables report the mean (standard deviation) over all 30 runs. Wilcoxon tests and Friedman ranks are both conducted at a significance level of $0.05$.

# 6 Results

## 6.1 Recovery of Governing Terms

MINT shows its clearest advantage on advection (Figure 2). It recovers the correct terms in $26/30$ runs on A2 and $13/30$ on A3, while Plain GP and SP-Prune recover neither equation. This advantage

**(a) Tree and PINN methods**

| | H2 | H3 | A2 | A3 | Burg. | Fish. |
|---|---|---|---|---|---|---|
| MINT (primary) | 3 | 0 | 26 | 13 | 0 | 0 |
| MINT-M (motif) | 4 | 0 | 29 | 8 | 17 | 19 |
| Plain GP | 10 | 0 | 0 | 0 | 0 | 3 |
| SP-Prune | 10 | 0 | 0 | 0 | 0 | 3 |
| Inverse PINN | Given | Given | Given | Given | Given | Given |
| Staged inverse | Given | Given | Given | Given | Given | Given |

**(b) Library methods**

| | H2 | H3 | A2 | A3 | Burg. | Fish. |
|---|---|---|---|---|---|---|
| DLGA-style mask | 22 | 0 | 30 | 0 | 30 | 0 |
| PDE-FIND-style | 0 | 0 | 30 | 11 | 30 | 2 |
| Weak regression | 5 | 0 | 0 | 0 | 30 | -- |
| DeepMoD-style | 0 | 0 | 30 | 26 | 30 | 2 |
| Linear Hybrid | 30 | 3 | 30 | 30 | -- | -- |

0 15 30 Given N/A

Figure 2: Exact term recoveries out of 30 noise-free runs, using a coefficient threshold of $10^{-4}$. "Given" means that the true active terms were supplied; – denotes N/A.

does not extend to every PDE family. On H2, both Plain GP and SP-Prune recover the correct terms in $10/30$ runs, compared with $3/30$ for MINT. On Fisher, both recover $3/30$, compared with none for MINT. H3 remains difficult across methods, with only the linear Hybrid achieving any recovery.

Methods supplied with a complete candidate library perform well on several benchmarks. All four library-based methods applicable to Burgers recover its terms in $30/30$ runs, whereas MINT recovers none. DeepMoD-style also achieves $30/30$ on A2 and $26/30$ on A3. These results show the value of restricting the search when the required terms are already available. The broader symbolic search remains difficult within the tested budget.

Adding family-specific motifs changes the nonlinear results. MINT-M recovers Burgers in $17/30$ runs and Fisher in $19/30$, showing that these initial expressions help the search. We report these results separately because MINT-M receives more prior information than MINT.

### 6.2 Field Accuracy

Table 2 compares the field accuracy of both MINT and MINT-M against each baseline, with mean errors and significance tests. Both variants significantly improve on the shared frozen reconstruction on A2 and A3, while MINT-M also improves Burgers and Fisher. MINT-M additionally outperforms DeepMoD-style on A2 and Fisher, although both variants have higher field errors than DeepMoD-style on A3 and Burgers. Notably, accurate field reconstruction does not necessarily accompany recovery of the governing terms: MINT achieves median relative field errors of $0.164\%$ on Burgers and $0.232\%$ on Fisher despite recovering neither equation, and on advection its median field-error ratio relative to the shared frozen reconstruction is 0.93 for A2 and rounds to 1.00 for A3. The large differences in term recovery therefore occur alongside much smaller changes in field accuracy. Compared with DeepMoD-style, MINT has lower field errors on H2 and H3, with median ratios of 0.59 and 0.96, but higher errors on A3 and Burgers, with ratios of 1.26 and 1.16. Table 2 reports the field-error summaries. We interpret statistical tests alongside the size of each difference: a ratio that rounds to 1.00 can have $p < 0.05$ while indicating little practical change.

### 6.3 Missing-Library Analysis

We remove $uu_x$ from the Burgers libraries and $u^2$ from the Fisher libraries, keeping all other settings and observations unchanged. GP-based methods retain their original grammar; MINT-M and given-support methods are excluded. Appendix F provides the full results and verifies that the removed terms are absent.

On Burgers, all four library methods drop from $30/30$ recoveries to zero, with mean residual ratios ranging from 353 to 7,552. On Fisher, however, DeepMoD-style consistently substitutes $u^3$ and achieves a mean residual ratio of 0.67 despite zero recovery. This comparison involves separately trained models, not the reference PDE evaluated with exact derivatives. Removing $u^2$ changes both term selection and the joint fitting of fields and coefficients. On these traveling-wave data, the substitute term and adjusted fields may compensate for derivative errors. The lower residual therefore indicates closer agreement between the learned field and equation, not recovery of the

| Method | H2 | H3 | A2 | A3 | Burgers | Fisher | Rank |
|---|---|---|---|---|---|---|---|
| MINT | 7.85 (1) | 50.4 (3.2) | 5.36 (1.2) | 11.5 (6.5) | 0.23 (0.24) | 0.279 (0.14) | **3.500** |
| MINT-M | 7.83 (0.96) | 50.4 (3.2) | 5.36 (1.2) | 11.6 (6.3) | 0.164 (0.041) | 0.22 (0.026) | **3.167** |
| Plain GP | 7.87 (1.1)$^{\approx/\approx}$ | 50.4 (3.2)$^{\approx/\approx}$ | 5.76 (1.3)$^{\uparrow/\uparrow}$ | 12 (6.1)$^{\uparrow/\uparrow}$ | 0.238 (0.23)$^{\approx/\uparrow}$ | 0.301 (0.18)$^{\approx/\uparrow}$ | 6.417 |
| SP-Prune | 7.87 (1.1)$^{\approx/\approx}$ | 50.4 (3.2)$^{\approx/\approx}$ | 5.76 (1.3)$^{\uparrow/\uparrow}$ | 12 (6.1)$^{\uparrow/\uparrow}$ | 0.238 (0.23)$^{\approx/\uparrow}$ | 0.301 (0.18)$^{\approx/\uparrow}$ | 6.417 |
| DLGA-style mask | 7.87 (1.1)$^{\approx/\approx}$ | 50.4 (3.2)$^{\approx/\approx}$ | 5.76 (1.3)$^{\uparrow/\uparrow}$ | 12 (6.1)$^{\uparrow/\uparrow}$ | 0.238 (0.23)$^{\approx/\uparrow}$ | 0.301 (0.18)$^{\approx/\uparrow}$ | 6.417 |
| PDE-FIND-style | 7.87 (1.1)$^{\approx/\approx}$ | 50.4 (3.2)$^{\approx/\approx}$ | 5.76 (1.3)$^{\uparrow/\uparrow}$ | 12 (6.1)$^{\uparrow/\uparrow}$ | 0.238 (0.23)$^{\approx/\uparrow}$ | 0.301 (0.18)$^{\approx/\uparrow}$ | 6.417 |
| Weak regression | 7.87 (1.1)$^{\approx/\approx}$ | 50.4 (3.2)$^{\approx/\approx}$ | 5.76 (1.3)$^{\uparrow/\uparrow}$ | 12 (6.1)$^{\uparrow/\uparrow}$ | 0.238 (0.23)$^{\approx/\uparrow}$ | – | – |
| DeepMoD-style | 13 (0.95)$^{\uparrow/\uparrow}$ | 51.8 (1.7)$^{\uparrow/\uparrow}$ | 5.43 (1.2)$^{\approx/\uparrow}$ | 8.17 (2.7)$^{\downarrow/\downarrow}$ | 0.144 (0.03)$^{\downarrow/\downarrow}$ | 0.271 (0.054)$^{\approx/\uparrow}$ | 4.667 |
| Linear Hybrid | 6.13 (0.5)$^{\downarrow/\downarrow}$ | 48 (2.3)$^{\downarrow/\downarrow}$ | 5.02 (1.2)$^{\downarrow/\downarrow}$ | 6.79 (1.7)$^{\downarrow/\downarrow}$ | – | – | – |
| Inverse PINN | 7.45 (0.86)$^{\approx/\approx}$ | 57.4 (2.4)$^{\uparrow/\uparrow}$ | 4.42 (1.2)$^{\downarrow/\downarrow}$ | 5.28 (1.1)$^{\downarrow/\downarrow}$ | 0.218 (0.21)$^{\approx/\approx}$ | 0.321 (0.24)$^{\approx/\approx}$ | 4.167 |
| Staged inverse | 7.87 (1.1)$^{\approx/\approx}$ | 50.4 (3.2)$^{\approx/\approx}$ | 5.46 (1.2)$^{\uparrow/\uparrow}$ | 11.5 (6.5)$^{\approx/\approx}$ | 0.138 (0.018)$^{\downarrow/\downarrow}$ | 0.214 (0.019)$^{\downarrow/\approx}$ | 3.833 |

Table 2: Relative $L_2$ field error (%): mean (std) over all 30 noise-free runs; lower is better. Wilcoxon tests with $p < 0.05$ are conducted, and superscripts refer to **MINT / MINT-M**: $\uparrow$ indicates significantly lower (better) error for the proposed method, $\downarrow$ higher (worse) error, and $\approx$ no significant difference. Friedman ranks average across all six PDEs with $p < 0.05$; weak regression and linear Hybrid are excluded because of N/A cases.

| $N_o$ | Noise | Variant | Shared field | Weak regression | DeepMoD-style | Linear Hybrid | Inverse PINN | Staged inverse |
|---|---|---|---|---|---|---|---|---|
| 100 | 1% | MINT | 1/5/0 | 0/5/0 | 1/0/5 | 0/0/4 | 0/0/6 | 0/4/2 |
| | | MINT-M | 2/4/0 | 1/4/0 | 1/1/4 | 0/0/4 | 0/1/5 | 0/5/1 |
| | 5% | MINT | 0/6/0 | 0/5/0 | 0/1/5 | 0/0/4 | 0/0/6 | 0/5/1 |
| | | MINT-M | 0/6/0 | 0/5/0 | 0/1/5 | 0/0/4 | 0/0/6 | 0/6/0 |
| 250 | 1% | MINT | 3/3/0 | 2/3/0 | 1/1/4 | 0/1/3 | 0/1/5 | 1/3/2 |
| | | MINT-M | 3/3/0 | 2/3/0 | 1/2/3 | 0/1/3 | 0/2/4 | 1/4/1 |
| | 5% | MINT | 3/3/0 | 2/3/0 | 1/0/5 | 0/0/4 | 0/0/6 | 0/4/2 |
| | | MINT-M | 3/3/0 | 2/3/0 | 1/1/4 | 0/0/4 | 0/1/5 | 0/6/0 |

Table 3: Field-accuracy **wins / nonsignificant differences / losses** for MINT and MINT-M against each baseline at 1% and 5% noise. Counts are across applicable PDEs, using 30 runs per PDE and Wilcoxon tests ($p < 0.05$).

governing law. MINT can express both removed terms but still recovers neither nonlinear equation. The results reveal two obstacles: a fixed library can exclude a required term, while a flexible search can fail to find it. Table S1 reports recovery counts and residual changes.

### 6.4 Noise and Observation Count

We report results for MINT and MINT-M at 1% and 5% noise, with 100 or 250 observations per trajectory. Noise is scaled by the standard deviation of clean field values in a fixed 500-point pool; only the first 100 or 250 noisy samples are given to the learner. Appendix G gives the detailed results. With 250 observations, MINT and MINT-M both achieve significantly lower field errors than the shared frozen reconstruction on A2, Burgers, and Fisher at 1% and 5% noise, with no significant losses on the other PDEs (Table 3). At 5% noise, the mean Fisher errors are $0.871\%$ for MINT and $0.761\%$ for MINT-M, compared with $1.10\%$ for the shared field. The benefit is less consistent with 100 observations: MINT and MINT-M improve on one and two PDEs, respectively, at 1% noise, and neither differs significantly from the shared fields at 5%.

Neither proposed method significantly outperforms the linear Hybrid on its four applicable PDEs, and inverse PINNs often give lower field errors. These comparisons involve different structural inputs (Table 1). The Hybrid receives a complete linear library, while standard and staged inverse PINNs receive the true active terms and estimate their coefficients. Restricting the candidate functions or fixing the support can stabilize estimation when noise makes the governing structure harder to identify. Their lower errors are therefore valuable results for settings where this information is available. MINT instead constructs composite terms within its grammar. The methods also differ in optimization, so the error gaps cannot be attributed to prior information alone.

The comparison with Plain GP and SP-Prune more directly assesses the benefit for symbolic discovery. With the same grammar and evolutionary budget, MINT recovers A2 more often in all four

noisy conditions without family-specific motifs. With 250 observations, MINT achieves $17/30$ and $6/30$ recoveries at 1% and 5% noise, respectively, while MINT-M achieves $22/30$ and $12/30$. Each GP-based baseline achieves $1/30$ at either noise level. These gains show that MINT can improve recovery within a flexible search space without requiring a complete library of composite terms.

MINT-M shows how family knowledge can further guide this search. At 250 observations and 5% noise, it recovers Burgers in $9/30$ runs and Fisher in $3/30$, compared with $1/30$ and $0/30$ for MINT. Its motifs provide additional structural information and may contain the correct form, but GP can modify or discard them rather than retaining a prescribed support. MINT-M therefore offers a way to use available family knowledge while preserving the ability to construct other expressions. Recovery remains sensitive to noise: its Fisher count falls from $22/30$ at 1% noise to $3/30$ at 5%.

Increasing the observation count from 100 to 250 lowers field error for both MINT and MINT-M in all 12 PDE–noise comparisons per method. For MINT-M, A2 recovery at 5% noise rises from $5/30$ to $12/30$, whereas MINT remains at $6/30$. More observations therefore improve field reconstruction throughout this test, but their effect on term recovery depends on the PDE and search method.

## 7 Conclusion

We introduced MINT, which combines independent neural fields with a shared GP tree to discover explicit PDEs from multiple trajectories. Cross-trajectory consistency guides the search for a common equation, while a two-stage procedure separates discrete structure discovery from continuous refinement of the fields and coefficients. Unlike inverse PINNs with prescribed equation forms, MINT searches for both terms and coefficients. Unlike fixed-library methods, it can construct composite terms within a specified grammar. MINT-M adds family-specific motifs to the initial population, allowing prior knowledge to guide the same shared-equation search.

Across six analytic benchmarks with 30 runs, MINT recovers two- and three-dimensional advection in $26/30$ and $13/30$ noise-free runs, respectively, while neither matched GP-based method recovers either equation. MINT-M recovers Burgers in $17/30$ runs and Fisher in $19/30$, where MINT has no recoveries, demonstrating the value of motif initialization for nonlinear search. With 250 observations and 5% noise, MINT and MINT-M recover two-dimensional advection in $6/30$ and $12/30$ runs, compared with $1/30$ for each GP-based baseline. These results establish clear advantages over the matched GP-based methods under the tested conditions, rather than uniform superiority over methods supplied with complete libraries or exact active terms.

The missing-library and noise experiments further distinguish equation recovery from numerical fit: an incorrect equation can achieve a lower residual, and additional observations improve field accuracy more consistently than term recovery. Fixed-tree refinement can improve continuous estimates but cannot repair a missing term or certify the governing law. Our evidence is limited to analytic benchmarks, with only traveling waves in the nonlinear cases. The gains from motif initialization point to a promising direction: learning reusable structural knowledge from related PDEs. Future work will explore transfer learning and multitask learning to share this knowledge across problems, guiding symbolic search and improving equation discovery. This would extend MINT from identifying a common equation across multiple trajectories of a single PDE to learning from related governing laws.

## AI use statement

The authors acknowledge the use of generative AI for language and grammar refinement, methodological discussion, coding, analysis, and manuscript preparation. The authors remain responsible for the final content.

## Reproducibility Statement

Appendices A–C describe the data partitions, sampling procedures, and analytic solutions. Appendix E documents the baseline adaptations, while Appendices F–G report supplementary results. We plan to open-source the code upon acceptance. Saved random seeds allow the data sets to be regenerated.

## REFERENCES


Mark J. Ablowitz and Anthony Zeppetella. Explicit solutions of fisher's equation for a special wave speed. *Bulletin of Mathematical Biology*, 41:835–840, 1979. doi: 10.1007/BF02462380.

Luca Biggio, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Parascandolo. Neural symbolic regression that scales. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 936–945, 2021.

Gert-Jan Both, Subham Choudhury, Pierre Sens, and Remy Kusters. DeepMoD: Deep learning for model discovery in noisy data. *Journal of Computational Physics*, 428:109985, 2021. doi: 10.1016/j.jcp.2020.109985.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016. doi: 10.1073/pnas.1517384113.

J. M. Burgers. A mathematical model illustrating the theory of turbulence. In *Advances in Applied Mechanics*, pp. 171–199. Elsevier, 1948. doi: 10.1016/S0065-2156(08)70100-5.

Lulu Cao, Zimo Zheng, Chenwen Ding, Jinkai Cai, and Min Jiang. Genetic programming symbolic regression with simplification-pruning operator for solving differential equations. In *Neural Information Processing*, Communications in Computer and Information Science, pp. 287–298. Springer Nature Singapore, 2023. doi: 10.1007/978-981-99-8132-8_22.

Zhao Chen, Yang Liu, and Hao Sun. Physics-informed learning of governing equations from scarce data. *Nature Communications*, 12:6136, 2021. doi: 10.1038/s41467-021-26434-1.

Miles Cranmer, Alvaro Sanchez-Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. In *Advances in Neural Information Processing Systems*, volume 33, pp. 17429–17442, 2020.

Lawrence C. Evans. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, 2 edition, 2010. doi: 10.1090/gsm/019.

R. A. Fisher. The wave of advance of advantageous genes. *Annals of Eugenics*, 7(4):355–369, 1937. doi: 10.1111/j.1469-1809.1937.tb02153.x.

Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002. doi: 10.1017/CBO9780511791253.

Daniel A. Messenger and David M. Bortz. Weak SINDy for partial differential equations. *Journal of Computational Physics*, 443:110525, 2021. doi: 10.1016/j.jcp.2021.110525.

Brenden K. Petersen, Mikel Landajuela Larma, T. Nathan Mundhenk, Claudio P. Santiago, Soo K. Kim, and Joanne T. Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations*, 2021.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

Samuel H. Rudy, Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017. doi: 10.1126/sciadv.1602614.

Hayden Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A*, 473(2197):20160446, 2017. doi: 10.1098/rspa.2016.0446.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009. doi: 10.1126/science.1165893.

Hao Xu, Haibin Chang, and Dongxiao Zhang. DLGA-PDE: Discovery of PDEs with incomplete candidate library via combination of deep learning and genetic algorithm. *Journal of Computational Physics*, 418:109584, 2020. doi: 10.1016/j.jcp.2020.109584.

# APPENDIX

This appendix provides the data construction, structural priors, and detailed results for MINT and MINT-M. It complements the main paper by separating term recovery from field accuracy and documenting the effects of motifs, missing library terms, noise, and observation count.

## A DATA PARTITIONS AND THEIR ROLES

For each trajectory $m$, we use four sets: labeled observations $\mathcal{D}_m$, discovery coordinates $\mathcal{C}_m$, checkpoint validation coordinates $\mathcal{V}_m$, and final evaluation coordinates $\mathcal{Q}_m$. Only $\mathcal{D}_m$ contains field labels. The coordinates $\mathcal{C}_m$, $\mathcal{V}_m$, and $\mathcal{Q}_m$ serve discovery and refinement, checkpoint selection, and final residual evaluation, respectively. Separate dense analytic grids measure field error and are excluded from fitting and selection.

### A.1 LABELED OBSERVATIONS

Observations take the form

$$y_{mi} = u^{(m)}(\mathbf{z}^D_{mi}) + \epsilon_{mi}. \tag{A.1}$$

The main experiment uses $N_o = 500$ noise-free observations. In the noise study, each trajectory and run has a fixed pool of 500 uniformly sampled coordinates and field values. We generate noise as

$$\epsilon_{mi} = \eta \sigma_{m,\mathrm{pool}} \xi_{mi}, \qquad \xi_{mi} \sim \mathcal{N}(0,1). \tag{A.2}$$

For the reported noise study, $\eta \in \{0.01, 0.05\}$; $\sigma_{m,\mathrm{pool}}$ is the population standard deviation of the pool values.

The learner receives the first $N_o \in \{100, 250\}$ noisy samples. Shared coordinates and Gaussian draws preserve pairing across conditions; each trajectory has an independent random stream. Scaling noise by the full pool keeps earlier samples unchanged as $N_o$ increases. Unused values are withheld, and the realized prefix noise level may differ from $\eta$. Uniform space–time sampling adds no dedicated boundary or initial-time points, and training uses no separate boundary or initial-condition losses. Refinement retains the observation loss.

### A.2 DISCOVERY COORDINATES

Each $\mathcal{C}_m$ contains $N_c = 2000$ scrambled Sobol points. Automatic differentiation of the neural fields supplies the candidate atoms $\phi_m$ and response $\partial_t u_{\theta_m}$; analytic derivatives are not used for training. The time derivative is excluded from the right-hand-side terminals. These coordinates support atom screening, GP fitness evaluation, constant fitting, and the refinement physics loss. Their residuals are training quantities, distinct from checkpoint and final-evaluation residuals.

### A.3 CHECKPOINT VALIDATION COORDINATES

Each $\mathcal{V}_m$ contains $N_v = 2000$ independently scrambled Sobol points, evaluated every $h_v$ refinement steps and at the final step. A checkpoint must satisfy the observation-fit and residual-improvement thresholds relative to the pre-refinement state; among eligible checkpoints, the lowest validation residual is retained. If none qualifies, the pre-refinement state is restored. These coordinates guide selection.

### A.4 FINAL EVALUATION COORDINATES

Each $\mathcal{Q}_m$ contains $N_q = 2000$ independent uniform points withheld from fitting and selection. After checkpoint restoration, MINT reports the mean squared residual across these points and trajectories, with the coefficient of $\partial_t u$ fixed to one. This residual combines equation and neural-derivative errors; field accuracy is evaluated separately. Appendix F explains why recorded residuals are not directly comparable across all methods.

# B Data Construction and Reproducibility

For run seed $s$ and trajectory $m \in \{1, \dots, M\}$, the sampling seeds are

$$s_m^D = s + 404 + 1009(m-1), \qquad s_m^C = s + 101 + 1009(m-1), \tag{B.1}$$

$$s_m^V = s + 700001 + 1009(m-1), \qquad s_m^Q = s + 900001 + 1009(m-1). \tag{B.2}$$

The offset $m-1$ corresponds to the zero-based trajectory index used in the implementation.

Discovery and validation coordinates use separately seeded PyTorch scrambled Sobol engines in float64. For $N$ points, we draw $2^{\lceil \log_2 N \rceil}$ points, retain the first $N$, and map them to the domain bounds. Uniform sampling and noise follow Appendix A. The generator verifies that the PDE coefficients are shared across trajectories and that trajectory parameters are distinct. Reproduction requires the saved seeds, trajectory formulas, domain bounds, sample counts, noise settings, and PyTorch version.

# C Trajectory Families

## C.1 Heat Equation

For the $d$-dimensional unit-diffusivity heat equation,

$$u_t = \sum_{j=1}^{d} u_{x_j x_j}, \tag{C.1}$$

the trajectory family is

$$H_{\mathbf{k}}(\mathbf{x}, t) = \exp\bigl(-\|\mathbf{k}\|_2^2 t\bigr) \prod_{j=1}^{d} \sin(k_j x_j). \tag{C.2}$$

Heat 2D uses $\mathbf{k} \in \{(1,1),(2,1),(1,2)\}$ on $[0,\pi]^2 \times [0,1]$. Heat 3D uses $\mathbf{k} \in \{(1,1,1),(2,1,1),(1,2,3),(3,1,2)\}$ on $[0,\pi]^3 \times [0,1]$. The diffusivity is shared across these modes, while their initial fields and decay rates vary with $\mathbf{k}$.

## C.2 Advection Equation

For advection with unit velocity in each coordinate direction,

$$u_t + \sum_{j=1}^{d} u_{x_j} = 0, \tag{C.3}$$

the plane-wave family is

$$A_{\mathbf{k}}(\mathbf{x}, t) = \sin\bigl(\mathbf{k}^{\mathsf{T}}\mathbf{x} - (\mathbf{1}^{\mathsf{T}}\mathbf{k})t\bigr). \tag{C.4}$$

Advection 2D uses $\mathbf{k} \in \{(1,0.5),(0.5,2),(1.5,1)\}$, and Advection 3D uses $\mathbf{k} \in \{(1,0.5,1.5),(0.5,2,1),(2,0.75,1.25)\}$, on $[0,2\pi]^d \times [0,1]$. Some frequencies are non-integer, so periodic boundary equality is neither assumed nor imposed.

## C.3 Viscous Burgers Equation

For

$$u_t = -uu_x + \nu u_{xx}, \qquad \nu = 0.1, \tag{C.5}$$

we use traveling shocks

$$B_{\ell,r,s}(x,t) = c - a\tanh\left(\frac{a(x - ct - s)}{2\nu}\right), \tag{C.6}$$

where $c = (\ell + r)/2$ and $a = (\ell - r)/2$. The three parameter tuples are $(1, -1, 0)$, $(1.5, 0.5, -0.6)$, and $(0.5, -1.5, 0.6)$ on $[-2, 2] \times [0, 1]$. The viscosity and nonlinear coefficient are shared across trajectories.

### C.4 Fisher–KPP Equation

For

$$u_t = Du_{xx} + \rho u(1-u), \qquad D = 0.1, \quad \rho = 1, \tag{C.7}$$

we use the analytic wave family

$$F_{s,d}(x,t) = \left[1 + \exp(\kappa(dx - ct - s))\right]^{-2}, \tag{C.8}$$

where $\kappa = \sqrt{\rho/(6D)}$, $c = 5\sqrt{D\rho/6}$, and $d \in \{-1, 1\}$ controls direction. The three pairs are $(s, d) \in \{(0, 1), (-1, -1), (1, 1)\}$ on $[-4, 4] \times [0, 1]$.

These families provide distinct fields with shared PDE coefficients: heat modes and transported plane waves (Evans, 2010; LeVeque, 2002), Burgers shocks, and Ablowitz–Zeppetella Fisher waves (Burgers, 1948; Fisher, 1937; Ablowitz & Zeppetella, 1979). Symbolic substitution verifies each trajectory; exact derivatives are excluded from training. The nonlinear benchmarks cover traveling waves, not general initial conditions.

## D Structural Information

### D.1 Terminal Screening

Let $\Phi_m \in \mathbb{R}^{N_c \times L}$ contain the learned time derivative $u_t$ and $L-1$ candidate atoms for trajectory $m$. Screening uses the stacked matrix

$$\Phi = [\Phi_1^\mathsf{T}, \ldots, \Phi_M^\mathsf{T}]^\mathsf{T}, \tag{D.1}$$

so all trajectories share one terminal set. The response $u_t$ is kept separately and excluded from the right-hand-side terminals.

After column normalization, an anchored sparse fit solves

$$u_t + \sum_j \beta_j \phi_j \approx 0 \tag{D.2}$$

and removes coefficients below a relative threshold. The retained atoms are combined with the top $k$ atoms ranked by

$$q_j = \frac{|\langle \phi_j, u_t \rangle|}{\|\phi_j\|_2 \|u_t\|_2}. \tag{D.3}$$

If fewer than three terms remain, including the response, the full declared atom set is restored. Screening selects primitive inputs, not composite terms.

### D.2 Family Motifs

MINT-M uses partial or complete family expressions, including products combined with derivatives and reactions such as $u - u^2$, to guide initialization. GP can modify or discard these seeds. MINT uses generic initialization. Because motifs may contain the correct structure, MINT-M evaluates the benefit of additional prior knowledge.

### D.3 Search-Space Assumptions

MINT receives the time coordinate, equation class, derivative order, atoms, operators, and tree constraints, but learns the active terms and coefficients. The formulation targets autonomous scalar evolution equations first order in time, with the coefficient of $u_t$ fixed to one. The grammar permits new atom combinations within this class; stationary, higher-order-in-time, coupled, or general implicit equations require a different formulation.

| Method | Burg.: full → missing | Residual ratio | Fish.: full → missing | Residual ratio |
|---|---|---|---|---|
| MINT | $0/30 \to 0/30$ | 1 (0) | $0/30 \to 0/30$ | 1 (0) |
| Plain GP | $0/30 \to 0/30$ | 1 (0) | $3/30 \to 3/30$ | 1 (0) |
| SP-Prune | $0/30 \to 0/30$ | 1 (0) | $3/30 \to 3/30$ | 1 (0) |
| DLGA-style mask | $30/30 \to 0/30$ | 653 (241) | $0/30 \to 0/30$ | 4.39 (0.262) |
| PDE-FIND-style | $30/30 \to 0/30$ | 618 (229) | $2/30 \to 0/30$ | 0.85 (0.462) |
| Weak regression | $30/30 \to 0/30$ | 7.55e+03 (3.79e+03) | – | – |
| DeepMoD-style | $30/30 \to 0/30$ | 353 (87) | $2/30 \to 0/30$ | 0.672 (0.358) |

Table S1: Recovery counts before and after term removal, with missing/full residual ratios reported as mean (std) over all 30 runs.

# E Baseline Implementations and Method Adaptations

All methods share observations and, where applicable, discovery coordinates within each PDE and run. Neural fields use three 64-unit $\tanh$ layers. The main paper's prior-information table distinguishes supplied priors from learned quantities; comparisons concern the following local adaptations. Plain GP searches expression trees for the PDE right-hand side (Schmidt & Lipson, 2009). SP-Prune adds small-constant pruning adapted from SP-GPSR, originally a solver for known PDEs (Cao et al., 2023). DLGA-style evolves masks over supplied columns rather than DLGA-PDE's derivative genes and products, so it cannot construct missing library terms (Xu et al., 2020). PDE-FIND-style applies STRidge to neural rather than numerical derivatives (Rudy et al., 2017); weak regression uses Monte Carlo weak features rather than structured-grid WSINDy (Messenger & Bortz, 2021). DeepMoD-style jointly fits neural fields and a sparse library model (Both et al., 2021). The linear Hybrid uses a linear library and is related to, but not a reproduction of, PINN-SR (Chen et al., 2021). Standard and staged inverse PINNs receive the active terms and estimate the remaining coefficients, fixing one coefficient to set the equation scale (Raissi et al., 2019).

# F Missing-Library Analysis

This study removes $uu_x$ on Burgers and $u^2$ on Fisher from applicable library methods, leaving GP grammars unchanged. All 390 completed runs and 30 N/A entries are present, with matching split identifiers. The 210 applicable library-removal runs confirm that excluded terms are absent from both the fitted libraries and active coefficients.

Removing the Burgers term reduces each applicable library method from $30/30$ recoveries to zero. On Fisher, DeepMoD-style instead achieves a lower residual despite failed recovery: its missing/full ratio is 0.672 (0.358), with median 0.70 (Table S1). MINT retains the ability to construct both removed interactions, an advantage over a library that excludes them. Its zero nonlinear recoveries nevertheless distinguish representational flexibility from search success. The separate MINT-M experiment shows how supplied motifs can help with the latter.

# G Noise and Observation Count

## G.1 Settings and Evaluation

The four conditions combine $N_o \in \{100, 250\}$ and $\eta \in \{0.01, 0.05\}$, giving 1,890 completed runs and 90 N/A entries in each. Weak regression on Fisher and linear Hybrid on both nonlinear PDEs remain N/A. Recovery follows the same $10^{-4}$ term criterion, counts non-polynomial outputs as failures, and excludes given-support PINNs. For each applicable method–PDE pair, we report the mean (standard deviation) over all 30 runs at each observation count and noise level. Wilcoxon signed-rank tests compare each MINT variant with each baseline at the $0.05$ level. Superscripts are ordered MINT/MINT-M: $\uparrow$ marks a significantly lower (better) error for the proposed method, $\downarrow$ a higher (worse) error, and $\approx$ no significant difference. We also run Friedman tests at each observation count and noise level and report the average ranks of all methods at the $0.05$ level. Weak regression and linear Hybrid are left out of the ranking because of N/A cases.

| Method | H2 | | | | H3 | | | |
|---|---|---|---|---|---|---|---|---|
| | $N_o = 100$ | | $N_o = 250$ | | $N_o = 100$ | | $N_o = 250$ | |
| | 1% | 5% | 1% | 5% | 1% | 5% | 1% | 5% |
| MINT | 0.0 | 0.0 | 26.7 | 16.7 | 0.0 | 0.0 | 0.0 | 0.0 |
| MINT-M | 0.0 | 0.0 | 26.7 | 3.3 | 0.0 | 0.0 | 0.0 | 0.0 |
| Plain GP / SP-Prune | 0.0 | 0.0 | 13.3 | 10.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| DLGA-style mask | 0.0 | 0.0 | 13.3 | 3.3 | 0.0 | 0.0 | 0.0 | 0.0 |
| PDE-FIND-style | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Weak regression | 0.0 | 0.0 | 23.3 | 10.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| DeepMoD-style | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Linear Hybrid | 63.3 | 63.3 | 100.0 | 96.7 | 0.0 | 0.0 | 6.7 | 10.0 |
| Inverse / staged inverse PINN | Given | Given | Given | Given | Given | Given | Given | Given |

Table S2: Exact term recovery (%) for H2 and H3 at 1% and 5% noise, with $N_o$ observations per trajectory. Each entry uses 30 runs per method. "Given" denotes supplied active terms.

| Method | A2 | | | | A3 | | | |
|---|---|---|---|---|---|---|---|---|
| | $N_o = 100$ | | $N_o = 250$ | | $N_o = 100$ | | $N_o = 250$ | |
| | 1% | 5% | 1% | 5% | 1% | 5% | 1% | 5% |
| MINT | 23.3 | 20.0 | 56.7 | 20.0 | 0.0 | 3.3 | 6.7 | 20.0 |
| MINT-M | 23.3 | 16.7 | 73.3 | 40.0 | 6.7 | 0.0 | 6.7 | 10.0 |
| Plain GP / SP-Prune | 0.0 | 3.3 | 3.3 | 3.3 | 0.0 | 0.0 | 0.0 | 0.0 |
| DLGA-style mask | 80.0 | 76.7 | 100.0 | 100.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| PDE-FIND-style | 3.3 | 0.0 | 80.0 | 36.7 | 0.0 | 0.0 | 0.0 | 0.0 |
| Weak regression | 0.0 | 0.0 | 0.0 | 3.3 | 0.0 | 0.0 | 0.0 | 0.0 |
| DeepMoD-style | 40.0 | 43.3 | 93.3 | 63.3 | 36.7 | 30.0 | 50.0 | 60.0 |
| Linear Hybrid | 100.0 | 100.0 | 100.0 | 100.0 | 66.7 | 66.7 | 86.7 | 90.0 |
| Inverse / staged inverse PINN | Given | Given | Given | Given | Given | Given | Given | Given |

Table S3: Exact term recovery (%) for A2 and A3 at 1% and 5% noise, with $N_o$ observations per trajectory. Each entry uses 30 runs per method. "Given" denotes supplied active terms.

### G.2 Term Recovery Under Noise

Tables S2–S4 compare H2/H3, A2/A3, and Burgers/Fisher side by side at 1% and 5% noise, with 100 or 250 observations per trajectory. Each entry uses 30 runs per method. "Given" and – denote supplied terms and N/A.

Both MINT variants recover A2 more often than Plain GP and SP-Prune at every reported noise level and observation count (Table S3). At 5% noise, MINT stays at $6/30$ as the observation count grows, whereas MINT-M rises from $5/30$ to $12/30$; each GP-based baseline stays at $1/30$. More observations thus help the motif-guided variant on this PDE, even though it starts one recovery behind MINT at 100 observations. On A3, DeepMoD-style and the linear Hybrid recover more often than either MINT variant in all four conditions. MINT-M improves Burgers recovery over MINT at both noise levels and observation counts (Table S4). With 250 observations and 5% noise, it recovers Burgers in $9/30$ runs and Fisher in $3/30$, against $1/30$ and $0/30$ for MINT. These gains come from the additional structural information the motifs provide. Recovery is still noise-sensitive: MINT-M's Fisher count drops from $22/30$ at 1% noise to $3/30$ at 5%. Several complete-library methods retain higher Burgers recovery.

### G.3 Field Accuracy

Tables S5 and S6 report field errors at 1% and 5% noise. Going from 100 to 250 observations lowers field error for both variants in all 12 PDE–noise comparisons per variant, so additional data benefits the neural fields consistently. With 250 observations, both variants also achieve significantly lower errors than the shared frozen reconstruction on A2, Burgers, and Fisher at both noise levels, with no significant losses on the other PDEs. At 5% noise, for instance, the mean Fisher error is $0.871\%$ for MINT and $0.761\%$ for MINT-M, against $1.10\%$ for the shared frozen field. At 100 observations and 5% noise, neither variant differs significantly from the shared fields, so the gain over the common data-only fit is clearest with the larger observation set.

| Method | Burgers | | | | Fisher–KPP | | | |
|---|---|---|---|---|---|---|---|---|
| | $N_o = 100$ | | $N_o = 250$ | | $N_o = 100$ | | $N_o = 250$ | |
| | 1% | 5% | 1% | 5% | 1% | 5% | 1% | 5% |
| MINT | 3.3 | 3.3 | 3.3 | 3.3 | 0.0 | 0.0 | 3.3 | 0.0 |
| MINT-M | 30.0 | 16.7 | 50.0 | 30.0 | 33.3 | 0.0 | 73.3 | 10.0 |
| Plain GP / SP-Prune | 3.3 | 3.3 | 3.3 | 3.3 | 3.3 | 0.0 | 3.3 | 0.0 |
| DLGA-style mask | 100.0 | 96.7 | 100.0 | 100.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| PDE-FIND-style | 96.7 | 3.3 | 96.7 | 13.3 | 3.3 | 0.0 | 3.3 | 3.3 |
| Weak regression | 100.0 | 20.0 | 100.0 | 66.7 | – | – | – | – |
| DeepMoD-style | 100.0 | 66.7 | 100.0 | 96.7 | 3.3 | 0.0 | 6.7 | 3.3 |
| Linear Hybrid | – | – | – | – | – | – | – | – |
| Inverse / staged inverse PINN | Given | Given | Given | Given | Given | Given | Given | Given |

Table S4: Exact term recovery (%) for Burgers and Fisher–KPP at 1% and 5% noise, with $N_o$ observations per trajectory. Each entry uses 30 runs per applicable method. "Given" denotes supplied active terms and – denotes N/A.

**(a)** $N_o = 100$**, noise** 1%

| Method | H2 | H3 | A2 | A3 | Burgers | Fisher | Rank |
|---|---|---|---|---|---|---|---|
| MINT | 25.5 (3.4) | 89.3 (7.2) | 24.6 (8.8) | 120 (14) | 0.574 (0.18) | 0.492 (0.26) | 5.333 |
| MINT-M | 25.5 (3.4) | 89.3 (7.2) | 24.5 (8.9) | 119 (15) | 0.519 (0.08) | 0.379 (0.078) | 4.000 |
| Plain GP | 25.5 (3.4)$^{\approx/\approx}$ | 89.3 (7.2)$^{\approx/\approx}$ | 25.3 (8.1)$^{\approx/\approx}$ | 121 (12)$^{\approx/\approx}$ | 0.574 (0.18)$^{\approx/\uparrow}$ | 0.566 (0.3)$^{\uparrow/\uparrow}$ | 6.667 |
| SP-Prune | 25.5 (3.4)$^{\approx/\approx}$ | 89.3 (7.2)$^{\approx/\approx}$ | 25.3 (8.1)$^{\approx/\approx}$ | 121 (12)$^{\approx/\approx}$ | 0.574 (0.18)$^{\approx/\uparrow}$ | 0.566 (0.3)$^{\uparrow/\uparrow}$ | 6.667 |
| DLGA-style mask | 25.5 (3.4)$^{\approx/\approx}$ | 89.3 (7.2)$^{\approx/\approx}$ | 25.3 (8.1)$^{\approx/\approx}$ | 121 (12)$^{\approx/\approx}$ | 0.574 (0.18)$^{\approx/\uparrow}$ | 0.566 (0.3)$^{\uparrow/\uparrow}$ | 6.667 |
| PDE-FIND-style | 25.5 (3.4)$^{\approx/\approx}$ | 89.3 (7.2)$^{\approx/\approx}$ | 25.3 (8.1)$^{\approx/\approx}$ | 121 (12)$^{\approx/\approx}$ | 0.574 (0.18)$^{\approx/\uparrow}$ | 0.566 (0.3)$^{\uparrow/\uparrow}$ | 6.667 |
| Weak regression | 25.5 (3.4)$^{\approx/\approx}$ | 89.3 (7.2)$^{\approx/\approx}$ | 25.3 (8.1)$^{\approx/\approx}$ | 121 (12)$^{\approx/\approx}$ | 0.574 (0.18)$^{\approx/\uparrow}$ | – | – |
| DeepMoD-style | 27.1 (3.2)$^{\uparrow/\uparrow}$ | 76.7 (4.8)$^{\downarrow/\downarrow}$ | 15 (4.2)$^{\downarrow/\downarrow}$ | 87.3 (13)$^{\downarrow/\downarrow}$ | 0.351 (0.11)$^{\downarrow/\downarrow}$ | 0.361 (0.093)$^{\downarrow/\approx}$ | 3.000 |
| Linear Hybrid | 15.7 (2.8)$^{\downarrow/\downarrow}$ | 84.1 (12)$^{\downarrow/\downarrow}$ | 13.1 (2.7)$^{\downarrow/\downarrow}$ | 95.6 (14)$^{\downarrow/\downarrow}$ | – | – | – |
| Inverse PINN | 15.9 (2.8)$^{\downarrow/\downarrow}$ | 74.7 (4)$^{\downarrow/\downarrow}$ | 9.96 (2.7)$^{\downarrow/\downarrow}$ | 67.1 (14)$^{\downarrow/\downarrow}$ | 0.394 (0.2)$^{\downarrow/\downarrow}$ | 0.428 (0.24)$^{\downarrow/\approx}$ | 1.667 |
| Staged inverse | 25.5 (3.4)$^{\approx/\approx}$ | 89.3 (7.2)$^{\approx/\approx}$ | 24.5 (8.8)$^{\approx/\approx}$ | 121 (12)$^{\approx/\approx}$ | 0.474 (0.11)$^{\downarrow/\downarrow}$ | 0.36 (0.071)$^{\downarrow/\approx}$ | 4.333 |

**(b)** $N_o = 250$**, noise** 1%

| Method | H2 | H3 | A2 | A3 | Burgers | Fisher | Rank |
|---|---|---|---|---|---|---|---|
| MINT | 11.6 (1.6) | 66 (4.6) | 7.38 (2.2) | 51.3 (13) | 0.35 (0.16) | 0.303 (0.12) | 5.417 |
| MINT-M | 11.6 (1.6) | 66 (4.6) | 7.38 (2.2) | 51.3 (13) | 0.266 (0.049) | 0.262 (0.033) | 4.750 |
| Plain GP | 11.6 (1.5)$^{\approx/\approx}$ | 66 (4.6)$^{\approx/\approx}$ | 8.05 (2.2)$^{\uparrow/\uparrow}$ | 51.3 (13)$^{\approx/\approx}$ | 0.376 (0.16)$^{\uparrow/\uparrow}$ | 0.321 (0.1)$^{\uparrow/\uparrow}$ | 6.333 |
| SP-Prune | 11.6 (1.5)$^{\approx/\approx}$ | 66 (4.6)$^{\approx/\approx}$ | 8.05 (2.2)$^{\uparrow/\uparrow}$ | 51.3 (13)$^{\approx/\approx}$ | 0.376 (0.16)$^{\uparrow/\uparrow}$ | 0.321 (0.1)$^{\uparrow/\uparrow}$ | 6.333 |
| DLGA-style mask | 11.6 (1.5)$^{\approx/\approx}$ | 66 (4.6)$^{\approx/\approx}$ | 8.05 (2.2)$^{\uparrow/\uparrow}$ | 51.3 (13)$^{\approx/\approx}$ | 0.376 (0.16)$^{\uparrow/\uparrow}$ | 0.321 (0.1)$^{\uparrow/\uparrow}$ | 6.333 |
| PDE-FIND-style | 11.6 (1.5)$^{\approx/\approx}$ | 66 (4.6)$^{\approx/\approx}$ | 8.05 (2.2)$^{\uparrow/\uparrow}$ | 51.3 (13)$^{\approx/\approx}$ | 0.376 (0.16)$^{\uparrow/\uparrow}$ | 0.321 (0.1)$^{\uparrow/\uparrow}$ | 6.333 |
| Weak regression | 11.6 (1.5)$^{\approx/\approx}$ | 66 (4.6)$^{\approx/\approx}$ | 8.05 (2.2)$^{\uparrow/\uparrow}$ | 51.3 (13)$^{\approx/\approx}$ | 0.376 (0.16)$^{\uparrow/\uparrow}$ | – | – |
| DeepMoD-style | 15.7 (1.1)$^{\uparrow/\uparrow}$ | 61.9 (2.4)$^{\downarrow/\downarrow}$ | 7.37 (2.2)$^{\approx/\approx}$ | 29.6 (8.4)$^{\downarrow/\downarrow}$ | 0.233 (0.041)$^{\downarrow/\downarrow}$ | 0.276 (0.04)$^{\downarrow/\approx}$ | 3.167 |
| Linear Hybrid | 7.8 (0.88)$^{\downarrow/\downarrow}$ | 60.7 (3.6)$^{\downarrow/\downarrow}$ | 7.02 (2.4)$^{\approx/\approx}$ | 25.1 (7.7)$^{\downarrow/\downarrow}$ | – | – | – |
| Inverse PINN | 9.19 (1.2)$^{\downarrow/\downarrow}$ | 62.6 (2.7)$^{\downarrow/\downarrow}$ | 6.18 (1.9)$^{\downarrow/\downarrow}$ | 13.7 (5.4)$^{\downarrow/\downarrow}$ | 0.299 (0.26)$^{\downarrow/\approx}$ | 0.33 (0.18)$^{\approx/\approx}$ | 3.000 |
| Staged inverse | 11.6 (1.5)$^{\approx/\approx}$ | 66 (4.6)$^{\approx/\approx}$ | 7.54 (2.2)$^{\uparrow/\uparrow}$ | 50.9 (14)$^{\approx/\approx}$ | 0.23 (0.035)$^{\downarrow/\downarrow}$ | 0.256 (0.02)$^{\downarrow/\approx}$ | 3.333 |

Table S5: Relative $L_2$ field error (%) at 1% noise: mean (std) over all 30 runs. Superscripts compare **MINT/MINT-M** with each baseline method; definitions and the nine-method ranking policy follow Section G.1. Friedman tests are also conducted. – denotes N/A or exclusion from ranking.

These gains do not solve every discovery problem. With 100 observations, MINT's H3 and A3 fields remain poorly reconstructed at both noise levels, so a small change in error between levels says little about reconstruction quality; the error level itself matters. Neither variant recovers H3 at any observation count or noise level, despite the improvement in field accuracy. More accurate fields alone do not ensure that the search identifies the governing terms. Complete-library and given-support methods still give lower errors on several PDEs, and the average ranks summarize these reconstruction differences, not the correctness of the recovered equations.

**(a)** $N_o = 100$**, noise** $5\%$

| Method | H2 | H3 | A2 | A3 | Burgers | Fisher | Rank |
|---|---|---|---|---|---|---|---|
| MINT | 27.2 (3.4) | 90.2 (7.5) | 29.2 (8.1) | 122 (12) | 3.27 (0.41) | 1.87 (0.25) | 5.167 |
| MINT-M | 27.2 (3.4) | 90.2 (7.5) | 29.2 (8.1) | 122 (12) | 3.27 (0.41) | 1.83 (0.31) | 5.167 |
| Plain GP | 27.2 (3.4)$^{\approx/\approx}$ | 90.2 (7.5)$^{\approx/\approx}$ | 29.5 (7.8)$^{\approx/\approx}$ | 122 (12)$^{\approx/\approx}$ | 3.27 (0.41)$^{\approx/\approx}$ | 1.91 (0.24)$^{\approx/\approx}$ | 6.333 |
| SP-Prune | 27.2 (3.4)$^{\approx/\approx}$ | 90.2 (7.5)$^{\approx/\approx}$ | 29.5 (7.8)$^{\approx/\approx}$ | 122 (12)$^{\approx/\approx}$ | 3.27 (0.41)$^{\approx/\approx}$ | 1.91 (0.24)$^{\approx/\approx}$ | 6.333 |
| DLGA-style mask | 27.2 (3.4)$^{\approx/\approx}$ | 90.2 (7.5)$^{\approx/\approx}$ | 29.5 (7.8)$^{\approx/\approx}$ | 122 (12)$^{\approx/\approx}$ | 3.27 (0.41)$^{\approx/\approx}$ | 1.91 (0.24)$^{\approx/\approx}$ | 6.333 |
| PDE-FIND-style | 27.2 (3.4)$^{\approx/\approx}$ | 90.2 (7.5)$^{\approx/\approx}$ | 29.5 (7.8)$^{\approx/\approx}$ | 122 (12)$^{\approx/\approx}$ | 3.27 (0.41)$^{\approx/\approx}$ | 1.91 (0.24)$^{\approx/\approx}$ | 6.333 |
| Weak regression | 27.2 (3.4)$^{\approx/\approx}$ | 90.2 (7.5)$^{\approx/\approx}$ | 29.5 (7.8)$^{\approx/\approx}$ | 122 (12)$^{\approx/\approx}$ | 3.27 (0.41)$^{\approx/\approx}$ | – | – |
| DeepMoD-style | 28 (3.2)$^{\approx/\approx}$ | 77 (4.7)$^{\downarrow/\downarrow}$ | 15.8 (3.1)$^{\downarrow/\downarrow}$ | 87.4 (12)$^{\downarrow/\downarrow}$ | 1.49 (0.22)$^{\downarrow/\downarrow}$ | 1.15 (0.16)$^{\downarrow/\downarrow}$ | 3.167 |
| Linear Hybrid | 16.4 (2.9)$^{\downarrow/\downarrow}$ | 84.3 (12)$^{\downarrow/\downarrow}$ | 15 (2.8)$^{\downarrow/\downarrow}$ | 96 (14)$^{\downarrow/\downarrow}$ | – | – | – |
| Inverse PINN | 16.1 (2.6)$^{\downarrow/\downarrow}$ | 74.9 (4)$^{\downarrow/\downarrow}$ | 12.2 (2.5)$^{\downarrow/\downarrow}$ | 71 (13)$^{\downarrow/\downarrow}$ | 1.28 (0.21)$^{\downarrow/\downarrow}$ | 1 (0.17)$^{\downarrow/\downarrow}$ | 1.000 |
| Staged inverse | 27.2 (3.4)$^{\approx/\approx}$ | 90.2 (7.5)$^{\approx/\approx}$ | 29.2 (8.1)$^{\approx/\approx}$ | 122 (12)$^{\approx/\approx}$ | 3.27 (0.41)$^{\approx/\approx}$ | 1.78 (0.37)$^{\downarrow/\approx}$ | 5.167 |

**(b)** $N_o = 250$**, noise** $5\%$

| Method | H2 | H3 | A2 | A3 | Burgers | Fisher | Rank |
|---|---|---|---|---|---|---|---|
| MINT | 12.4 (1.4) | 66.2 (4.5) | 9.32 (1.8) | 53.6 (12) | 1.45 (0.31) | 0.871 (0.25) | 4.667 |
| MINT-M | 12.5 (1.4) | 66.2 (4.5) | 9.37 (1.7) | 53.6 (12) | 1.35 (0.38) | 0.761 (0.18) | 5.000 |
| Plain GP | 12.5 (1.4)$^{\approx/\approx}$ | 66.2 (4.5)$^{\approx/\approx}$ | 9.82 (1.7)$^{\uparrow/\uparrow}$ | 53.6 (12)$^{\approx/\approx}$ | 1.56 (0.23)$^{\uparrow/\uparrow}$ | 1.1 (0.26)$^{\uparrow/\uparrow}$ | 6.583 |
| SP-Prune | 12.5 (1.4)$^{\approx/\approx}$ | 66.2 (4.5)$^{\approx/\approx}$ | 9.82 (1.7)$^{\uparrow/\uparrow}$ | 53.6 (12)$^{\approx/\approx}$ | 1.56 (0.23)$^{\uparrow/\uparrow}$ | 1.1 (0.26)$^{\uparrow/\uparrow}$ | 6.583 |
| DLGA-style mask | 12.5 (1.4)$^{\approx/\approx}$ | 66.2 (4.5)$^{\approx/\approx}$ | 9.82 (1.7)$^{\uparrow/\uparrow}$ | 53.6 (12)$^{\approx/\approx}$ | 1.56 (0.23)$^{\uparrow/\uparrow}$ | 1.1 (0.26)$^{\uparrow/\uparrow}$ | 6.583 |
| PDE-FIND-style | 12.5 (1.4)$^{\approx/\approx}$ | 66.2 (4.5)$^{\approx/\approx}$ | 9.82 (1.7)$^{\uparrow/\uparrow}$ | 53.6 (12)$^{\approx/\approx}$ | 1.56 (0.23)$^{\uparrow/\uparrow}$ | 1.1 (0.26)$^{\uparrow/\uparrow}$ | 6.583 |
| Weak regression | 12.5 (1.4)$^{\approx/\approx}$ | 66.2 (4.5)$^{\approx/\approx}$ | 9.82 (1.7)$^{\uparrow/\uparrow}$ | 53.6 (12)$^{\approx/\approx}$ | 1.56 (0.23)$^{\uparrow/\uparrow}$ | – | – |
| DeepMoD-style | 16.2 (1.1)$^{\uparrow/\uparrow}$ | 62.1 (2.4)$^{\downarrow/\downarrow}$ | 8.25 (1.9)$^{\downarrow/\downarrow}$ | 30.3 (7.1)$^{\downarrow/\downarrow}$ | 0.873 (0.15)$^{\downarrow/\downarrow}$ | 0.714 (0.075)$^{\downarrow/\approx}$ | 2.833 |
| Linear Hybrid | 8.04 (0.79)$^{\downarrow/\downarrow}$ | 61 (3.5)$^{\downarrow/\downarrow}$ | 7.95 (2.1)$^{\downarrow/\downarrow}$ | 28.2 (6.9)$^{\downarrow/\downarrow}$ | – | – | – |
| Inverse PINN | 9.24 (1.2)$^{\downarrow/\downarrow}$ | 62.6 (2.6)$^{\downarrow/\downarrow}$ | 6.95 (1.8)$^{\downarrow/\downarrow}$ | 16.6 (5)$^{\downarrow/\downarrow}$ | 0.754 (0.18)$^{\downarrow/\downarrow}$ | 0.728 (0.21)$^{\downarrow/\approx}$ | 1.333 |
| Staged inverse | 12.5 (1.5)$^{\approx/\approx}$ | 66.2 (4.5)$^{\approx/\approx}$ | 9.3 (1.8)$^{\approx/\approx}$ | 53.6 (12)$^{\approx/\approx}$ | 1.28 (0.41)$^{\downarrow/\approx}$ | 0.73 (0.08)$^{\downarrow/\approx}$ | 4.833 |

Table S6: Relative $L_2$ field error (%) at 5% noise: mean (std) over all 30 runs. MINT/MINT-M superscripts and the nine-method ranking policy follow Table S5. Friedman tests are also conducted. – denotes N/A or exclusion from ranking.