PHYSICS-INFORMED MEANFLOW FOR PDE-CONSTRAINED INVERSE PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models have demonstrated promising results for inverse problems governed by Partial Differential Equations (PDEs). However, current diffusion-based methods make limited progress at early sampling stages: constraints are evaluated on the Tweedie posterior-mean estimate, which at high noise is close to the dataset mean and therefore carries little information about the specific solution. Moreover, accurate reconstruction can require thousands of guided denoising steps, each depending on the previous one. Both effects slow convergence. To address them, we introduce **P**hysics-**I**nformed **M**ean**F**low (PIMF), which fine-tunes a MeanFlow prior with a PDE-residual loss and, at inference, evaluates observation and PDE constraints on the MeanFlow one-step estimate of the clean state rather than on the posterior mean, so that guidance is informative from the first step. PIMF (i) improves accuracy and computational efficiency over diffusion-based solvers on most benchmark tasks, (ii) makes ensemble-based uncertainty estimation affordable, and (iii) recovers alternative physically valid solutions consistent with the same sparse observations.

1 INTRODUCTION

Inverse problems (Schönlieb & Shumaylov, 2025; Ke & Schönlieb, 2021; Benning & Burger, 2018; Benning et al., 2014) require inferring unknown parameters, initial conditions, system properties, and recovering a complete solution from partial or noisy observations. This recovery is highly sensitive to noise and is ill-posed, meaning that partial observations can correspond to multiple valid solutions. Inverse problems arise in many applications, including medical imaging (Song et al., 2022), seismic inversion (Chen & Saygin, 2021), climate science (Kalnay, 2003), and chemistry (Jonas, 2019). In scientific computing, inverse problems involving Partial Differential Equations (PDEs) are particularly common, e.g., in fluid dynamics (Eckert, 2007) and thermodynamics (Borgnakke, 2025).

Generative models offer a natural framework for these problems by learning data-driven priors that regularize these ill-posed tasks. For general inverse problems, Diffusion Posterior Sampling (DPS) (Chung et al., 2023) first trains a Diffusion Model (DM) on common image data, and attempts to solve linear (inpainting, Gaussian deblur, super-resolution, and motion deblur) and nonlinear (phase retrieval and non-uniform deblur) problems by denoising according to the respective measurements, and approximating the corresponding posterior probability.

DiffusionPDE (Huang et al., 2024) extends DPS with PDE-residual guidance, steering the diffusion trajectory toward reconstructions that are both physically plausible and consistent with the observations. At each step, it evaluates the observation and PDE residuals on an estimate of the clean field and back-propagates these losses to the current intermediate state. Although it outperforms non-generative baselines, two limitations remain in the settings we study (Figure 1c): the reconstruction starts to match the observations only late in the trajectory, and reaching full accuracy can require thousands of guided steps, which makes every additional reconstruction (e.g., for an ensemble) expensive.

Diffusion sampling starts from Gaussian noise and denoises it progressively; we call the partially denoised field at any point of this process the *intermediate state*. Evaluating observation and PDE constraints directly on an intermediate state is uninformative, so diffusion-based guidance evaluates them on the expected clean field given that state, i.e., the posterior mean. During early stages of sampling, however, this estimate is close to the dataset mean (Karras et al., 2022; Daras et al., 2024):

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.

as noise dominates, the learned denoiser returns a global average that carries almost no information about the particular solution being reconstructed (Figure 1b).

One-step generative models such as MeanFlow (Geng et al., 2025) offer an alternative estimator. MeanFlow is trained to map any intermediate state to a plausible clean field in a single network call, but also can be run iteratively to increase its quality. Although in general a one-step MeanFlow generation is less accurate than a many-step diffusion one, its output is trained to resemble an individual clean field rather than an average over plausible fields, so constraints evaluated on it yield useful guidance from the very first step.

Our approach. Building on this, we introduce **P**hysics-**I**nformed **M**ean**F**low (PIMF), detailed in Section 4. Our contributions are:

1. **Physics-informed MeanFlow training** (Section 4.1, Figure 1a). We train a MeanFlow prior over joint coefficient–solution fields and then fine-tune it with a governing-residual penalty on its one-step clean-state estimate, creating an efficient Physics-informed MeanFlow (PIMF) model.
2. **Projection-based guidance** (Section 4.2). We evaluate observation losses on the MeanFlow estimate throughout reconstruction and add PDE-residual guidance during late refinement.
3. **Evaluation on five PDE families** (Section 5). PIMF surpasses DiffusionPDE in accuracy and computational efficiency on most benchmark tasks, and an ablation (Section 5.4) separates the contribution of training-time from inference-time physics.
4. **Uncertainty estimation and solution discovery** (Section 5.6). The saved budget makes ensemble-based uncertainty estimation (the empirical dispersion of independent reconstructions) practical at the cost of a single diffusion run, and the same ensemble recovers distinct solutions consistent with the sparse observations, which on Poisson and Helmholtz also nearly satisfy an independent forward solve (App. H).

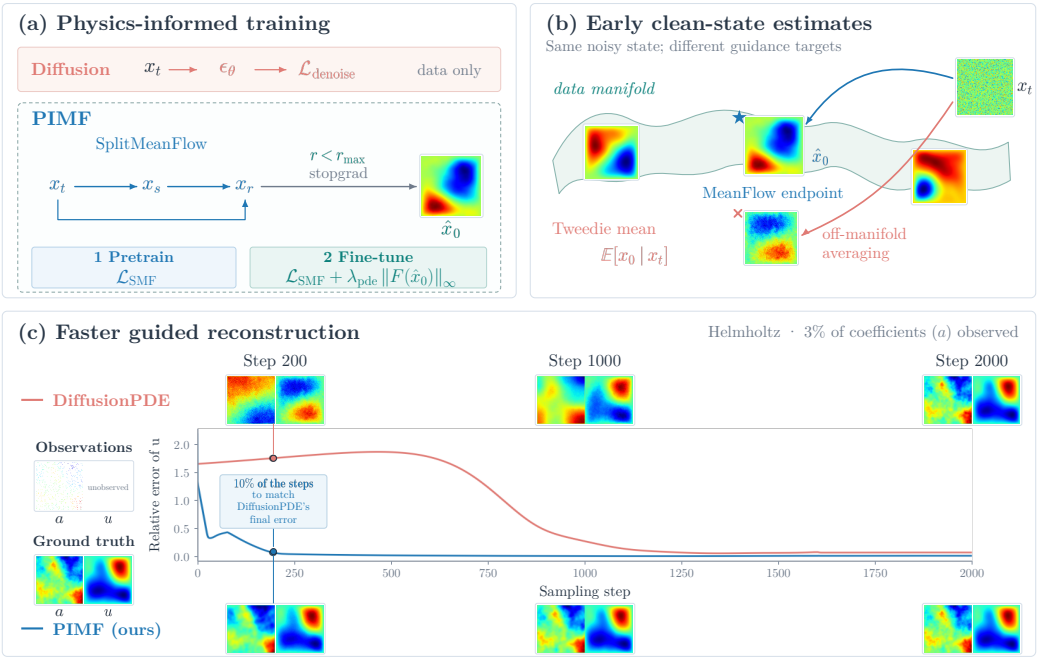

Figure 1: **PIMF vs. DiffusionPDE. a)** Training: diffusion minimizes a data-driven denoising loss, PIMF adds a PDE-residual penalty on its clean-state projection $\hat{x}_0$. **b)** Early inference: the Tweedie estimate leaves the data manifold while the MeanFlow projection remains close to it. **c)** Inference: On Helmholtz forward, PIMF reaches DiffusionPDE's final error in $\sim$10% of the steps.

2 PROBLEM FORMULATION

We solve forward and inverse problems in which one component of a PDE state is sparsely observed and both the coefficient a and solution field u must be recovered. Building on the benchmark suite established by Huang et al. (2024), we investigate two broad classes of PDEs.

Stationary Systems. For static problems, such as Darcy Flow or the Inhomogeneous Helmholtz equation, the physical system is defined by a time-independent function f:

$$f(c; a, u) = 0, \quad c \in \Omega \subset \mathbb{R}^d, \tag{1}$$

where c represents the spatial coordinates in the domain Ω, $a \in \mathcal{A}$ denotes the PDE medium property field (e.g., permeability or source term), and $u \in \mathcal{U}$ is the corresponding solution field. We assume a known boundary condition $u|_{\partial\Omega} = g(c)$ on the domain boundary $\partial\Omega$.

Time-Dependent Systems. For dynamic problems, such as the Navier-Stokes equations, we consider the evolution of a state u over a temporal interval τ[1] $\in [0, T]$:

$$f(c, \tau; a, u) = 0, \quad c \in \Omega, \tau \in (0, T], \tag{2}$$

where $a = u_0 \in \mathcal{A}$ denotes the initial condition at $\tau = 0$. The objective is to recover the initial state a and the solution snapshot at the final time $u_T := u(\cdot, T)$ from partial measurements. In the specific case of the one-dimensional Burgers' equation, this approach is extended to recover the full spatiotemporal trajectory $u_{0:T}$.

For the rest of the paper, we will use the term "coefficient" to refer to a, regardless of the problem's time-dependence. We assume $x = (a, u)$ is jointly distributed according to an unknown distribution $p(a, u)$, induced by sampling coefficients a from a prior distribution $p(a)$ and solving the PDE for each a to obtain the solution u. Learning this joint prior lets the same reconstruction procedure address forward and inverse tasks without differentiating through the numerical PDE solver; the governing equation is instead imposed through residual-based training and guidance.

3 BACKGROUND

3.1 SOLVING PDE-CONSTRAINED INVERSE PROBLEMS

Given observations $y = \mathcal{M}(x) + \eta$, where $\mathcal{M}$ is a known measurement operator and η is observation noise, we aim to sample from the posterior $p(x \mid y)$ using a learned generative model as prior. By Bayes' rule and noticing that $\nabla_x \log p(y) = 0$:

$$\nabla_x \log p(x \mid y) = \nabla_x \log p(x) + \nabla_x \log p(y \mid x).$$

Following Dhariwal & Nichol (2021), we consider sharpened posteriors $p_\zeta(x \mid y) \propto p(x) \cdot p(y \mid x)^\zeta$, giving the guided score:

$$\nabla_x \log p_\zeta(x \mid y) = \nabla_x \log p(x) + \zeta \nabla_x \log p(y \mid x), \tag{3}$$

where $\zeta = 1$ gives the untempered posterior score and $\zeta > 1$ concentrates probability on modes. Diffusion models (DMs) (Ho et al., 2020; Song et al., 2021) learn the score $\nabla_{x_t} \log p_t(x_t)$ at every noise level via denoising score matching, supplying the prior term of equation 3.

Diffusion Posterior Sampling (DPS) (Chung et al., 2023) approximates the intractable likelihood $p(y \mid x_t)$ at an intermediate state x_t via a denoised estimate $\hat{x}_0$ of x_t from Tweedie's formula. Under a Gaussian observation model, up to a positive scale absorbed into the guidance weight,

$$\nabla_{x_t} \log p(y \mid x_t) \approx -\nabla_{x_t} \|y - \mathcal{M}(\hat{x}_0)\|_2^2. \tag{4}$$

With an exact score, the Tweedie estimate equals $\mathbb{E}[x_0 \mid x_t]$; a learned denoiser approximates this conditional mean.

DiffusionPDE (Huang et al., 2024) builds on top to target PDE-constrained inverse problems by learning the joint distribution $p(a, u)$ over PDE coefficients a and solution fields u. During inference, it adds a PDE residual guidance term alongside the observation term, $\mathcal{L}_{\text{pde}} = \|\mathcal{F}(\hat{x}_0)\|_2$, penalizing

[1] We denote physical time by τ and diffusion-related time by t.

violations of governing equations $\mathcal{F}(a, u) = 0$. Its released implementation differentiates unsquared norms for both the observation and the PDE term, and we follow this convention for every guided method in this paper. The combined guided score is:

$$\nabla_{x_t} \log p_{\boldsymbol{\zeta}}(x_t|y) \approx \nabla_{x_t} \log p_t(x_t) - \zeta_{\text{obs}} \nabla_{x_t} \mathcal{L}_{\text{obs}} - \zeta_{\text{pde}} \nabla_{x_t} \mathcal{L}_{\text{pde}},$$

where $\zeta_{\text{obs}}, \zeta_{\text{pde}}$ are per-term guidance weights of the form Equation 3. The resulting dynamics induce a heuristic conditional distribution rather than exact posterior sampling (Lai et al., 2025).

3.2 Flow Matching and MeanFlow

Flow Matching (FM) (Lipman et al., 2023; 2024) learns a time-dependent vector field $v_\theta(x, t)$ to transport samples from a prior $x_1 \sim \mathcal{N}(0, I)$ to the data distribution $x_0 \sim p_{\text{data}}$ via the ODE $\frac{dx_t}{dt} = v_\theta(x_t, t)$. Under a Gaussian prior and an independent coupling between prior and data, Flow Matching and the diffusion process of Section 3.1 describe the same stochastic interpolant (Albergo et al., 2025). This interpolant motivates using the same observation and PDE losses to guide the velocity trajectory. Additionally, due to a simpler noise schedule, it has fewer hyperparameters to tune. Training minimizes a regression objective on linearly-interpolated paths $x_t = (1 - t)x_0 + tx_1$:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t,x_0,x_1}\left[\|v_\theta(x_t, t) - (x_1 - x_0)\|_2^2\right].$$

MeanFlow. Flow Matching requires many small steps to discretize a continuous trajectory. One-step methods (Zhou et al., 2025; Song et al., 2023b; Kim et al., 2024; Frans et al., 2025; Song & Dhariwal, 2024) avoid this; MeanFlow (Geng et al., 2025) models the mean velocity over a finite time interval, making it suitable for large-step evaluation. For $r < t$,

$$m(x_t, r, t) \triangleq \frac{1}{t - r} \int_r^t v(x_s, s)\, \mathrm{d}s, \qquad \lim_{r \to t} m(x_t, r, t) = v(x_t, t). \tag{5}$$

A network m_θ learns this quantity using

$$\mathcal{L}_{\text{MF}}(\theta) = \mathbb{E}\, \tfrac{1}{2}\|m_\theta(x_t, r, t) - \text{sg}(m_{\text{tgt}})\|_2^2,$$

where sg denotes stop-gradient and the original MeanFlow target is

$$m_{\text{tgt}} = v(x_t, t) - (t - r)\big(v(x_t, t)\, \partial_x m_\theta + \partial_t m_\theta\big).$$

Computing this target requires a Jacobian–vector product. SplitMeanFlow (Guo et al., 2025) avoids it by exploiting the interval-splitting consistency of the mean velocity: for any $s \in [r, t]$, $(t - r)\, m(x_t, r, t) = (t - s)\, m(x_t, s, t) + (s - r)\, m(x_s, r, s)$ with $x_s = x_t - (t - s)\, m(x_t, s, t)$. Writing $s = (1 - \gamma)t + \gamma r$ with $\gamma \sim \mathcal{U}[0, 1]$ gives the alternative target

$$m_{\text{tgt}}^{\text{SMF}} = (1 - \gamma)\, m_\theta(x_s, r, s) + \gamma\, m_\theta(x_t, s, t), \qquad \mathcal{L}_{\text{SMF}}(\theta) = \mathbb{E}\, \tfrac{1}{2}\|m_\theta(x_t, r, t) - \text{sg}(m_{\text{tgt}}^{\text{SMF}})\|_2^2,$$

which only needs two additional gradient-free network evaluations. We use $\mathcal{L}_{\text{SMF}}$ throughout.

Sampling proceeds as $x_{t_{i+1}} = x_{t_i} - \Delta t \cdot m_\theta(x_{t_i}, t_{i+1}, t_i)$ for N steps from $t_0 = 1$ to $t_N = 0$ with $\Delta t = 1/N$. At any timestep t, we obtain the clean state estimate via the one-step projection:

$$\hat{x}_0 = x_t - t\, m_\theta(x_t, 0, t). \tag{6}$$

We refer to this map as the MeanFlow *projection*: its output is a learned estimate of an individual clean field rather than an average over plausible ones. Both the Tweedie and MeanFlow estimates are deterministic maps for a fixed x_t, but they differ in their definition. With an exact score, Tweedie's formula gives $\mathbb{E}[x_0 \mid x_t]$, whereas MeanFlow predicts the endpoint implied by its learned finite-horizon velocity and is not defined as a conditional mean.

4 Physics-Informed MeanFlow

To overcome the limitations of diffusion-based approaches, we introduce **P**hysics-**I**nformed **M**ean**F**low (PIMF). Our method comprises two parts: a physics-informed generative model trained on the joint distribution $p(a, u)$, and an algorithm that uses the model's predictions to recover a full solution from observations.

4.1 Training a Physics-Informed MeanFlow

To build a generative prior that respects physical laws, we train a MeanFlow model on the joint distribution $p(a, u)$ using a staged procedure. Naively training the model with a PDE residual penalty carries two issues: (i) PDE gradients are unreliable when the velocity field is poorly trained, and backpropagating through them can destabilize the training, and (ii) enforcing PDE constraints from the beginning can cause the model to collapse to trivial minimizers (e.g., constant fields), rather than learning the true data distribution. To address these, we train the model in two stages.

Stage 1: (Standard) MeanFlow pretraining. For the first half of training, we minimize the SplitMeanFlow loss $\mathcal{L}_{\mathrm{SMF}}$ of Section 3.2. We use a linear probability path $x_t = (1-t)x_0 + tx_1$, with $x_1 \sim \mathcal{N}(0, I)$, $x_0 \sim p_{\mathrm{data}}$, and $t \sim \mathcal{U}[0, 1]$. With probability 0.6, we set $r = t$; otherwise, we sample $r \sim \mathcal{U}[0, t]$, following Geng et al. (2025). The complete target construction is given in Algorithm 2.

Stage 2: Physics-informed fine-tuning. For the second half of training, we add a PDE residual penalty evaluated on a projection to the data endpoint of the state x_r reached inside the SplitMeanFlow loss (Figure 1a). On the interval-splitting branch, whenever the sampled interval start satisfies $r < r_{\max}$, we reuse the branch's prediction $m_\theta(x_t, r, t)$ to hop to $x_r = x_t - (t-r)\, m_\theta(x_t, r, t)$ and project to the endpoint under a stop-gradient:

$$\hat{x}_0 = x_r - r\,\mathrm{sg}[m_\theta(x_r, 0, r)]\,, \qquad \mathcal{L}_{\mathrm{total}}(\theta) = \mathcal{L}_{\mathrm{SMF}}(\theta) + \lambda_{\mathrm{pde}}\,\mathbb{E}[\mathbf{1}\{r < r_{\max}\}\,\|\mathcal{F}(\hat{x}_0)\|_\infty]\,, \quad (7)$$

where $\mathrm{sg}[\cdot]$ denotes stop-gradient. Gradients therefore reach θ only through the first hop, which reuses the forward pass of the $\mathcal{L}_{\mathrm{SMF}}$ splitting branch, while the gate $r < r_{\max}$ restricts the penalty to states near the data endpoint, where the projection is accurate. We use the L^∞ norm for the residual following Wang et al. (2022); Chang et al. (2025), who show that penalizing the worst-case residual, rather than the mean-squared residual, is beneficial in physics-informed training (Fig. 4).

A MeanFlow model can be trained either from scratch or initialized from a pretrained Flow Matching model (Geng et al., 2025). We evaluate both approaches, denoted **PIMF** and **PIMF-FMinit**, respectively. Initialization improves the model's performance and convergence, at the cost of substantially more total training iterations used for the base model. We provide full architectural and training details in App. A.

4.2 Guided MeanFlow Inference

Notation. We consider a (decreasing) time discretization $\{t_i\}_{i=0}^N$ and denote $x_i := x_{t_i}$ for readability.

Projection-based guidance. For inverse problems with sparse observations, we perform sampling from $t_0 = 1$ to $t_N = 0$. At each step t_i with current state x_i, we compute the MeanFlow projection to obtain $\hat{x}_0$, and evaluate observational and PDE constraints on this projected state. We combine two objectives: an observation loss over the observed channels and a PDE residual loss $\|\mathcal{F}(\hat{x}_0)\|_2$.

Staged guidance schedule. Following the DiffusionPDE protocol, observation guidance is active throughout reconstruction. The schedule uses two hyperparameters: δ_{pde} specifies the normalized iteration at which PDE guidance is activated, and α_{obs} rescales the observation contribution after that transition. We use the same setting for every guided method; numerical values and dataset-specific weights are listed in App. C.

5 Experiments

5.1 Experimental Setup and PDE Families

To evaluate PIMF, we adopt the benchmark setup of Huang et al. (2024), and we reuse their datasets. We sample coefficient fields (for stationary problems) and initial vorticity fields (for Navier–Stokes) from Gaussian random fields and evolve them with finite-difference or spectral solvers, respectively. We use identical parameter configurations to make the comparison with existing solvers as rigorous and transparent as possible.

We evaluate on five PDE families: three stationary elliptic systems (Darcy Flow, Helmholtz, and Poisson) and two time-evolutionary dynamics (Navier–Stokes on a non-bounded domain, NSNB, and

Algorithm 1 Guided Physics-Informed MeanFlow inference

Require: Trained m_θ; grid $1 = t_0 > \cdots > t_N = 0$; observations $\{(y_c, \mathcal{M}_c)\}_{c\in\mathcal{C}}$; weights $\{\zeta_{\mathrm{obs},c}\}_{c\in\mathcal{C}}$ and ζ_{pde}; schedule hyperparameters δ_{pde} and α_{obs}

Sample $x_0 \sim \mathcal{N}(0, I)$; set $i_{\mathrm{pde}} \leftarrow \lfloor \delta_{\mathrm{pde}} N \rfloor$
for $i = 0, \ldots, N-1$ **do**
  $\hat{x}_0^{(i)} \leftarrow x_i - t_i m_\theta(x_i, 0, t_i)$
  $g_{\mathrm{obs}}^{(i)} \leftarrow \sum_{c\in\mathcal{C}} \zeta_{\mathrm{obs},c} \nabla_{x_i} \|\mathcal{M}_c(\hat{x}_{0,c}^{(i)}) - y_c\|_2$
  $\bar{x}_{i+1} \leftarrow x_i - (t_i - t_{i+1}) m_\theta(x_i, t_{i+1}, t_i)$
  if $i < i_{\mathrm{pde}}$ **then**
    $x_{i+1} \leftarrow \bar{x}_{i+1} - g_{\mathrm{obs}}^{(i)}$
  else
    $g_{\mathrm{pde}}^{(i)} \leftarrow \zeta_{\mathrm{pde}} \nabla_{x_i} \|\mathcal{F}(\hat{x}_0^{(i)})\|_2$
    $x_{i+1} \leftarrow \bar{x}_{i+1} - \alpha_{\mathrm{obs}} g_{\mathrm{obs}}^{(i)} - g_{\mathrm{pde}}^{(i)}$
  end if
end for
return x_N

Burgers); App. B provides the governing equations, guidance functions, and physical parameters. The first four datasets contain paired fields $x = (a, u)$ represented on 128×128 arrays, while Burgers is reconstructed as a complete spatiotemporal (single-channel) field.

Following Huang et al. (2024), we recover both a and u from 500 sparse observations available in only one of the two fields (for Burgers, five observed sensor columns of the space–time field, i.e., 640 points). The forward task observes a and reports the reconstruction error of u; the inverse task observes u and reports the error of a. Unless stated otherwise, we evaluate all methods using $N = 2000$ sampling steps on 1000 test instances per PDE.

For a target field q, we report the mean per-instance relative error $\mathrm{Rel}(\hat{q}, q) = \frac{100}{n} \sum_{j=1}^{n} \frac{\|\hat{q}_j - q_j\|_2}{\|q_j\|_2}$. Standard deviations in tables are across test instances; trajectory and budget curves show ± 1 standard error. Full results for both channels and all evaluation details appear in App. D.

5.2 Baselines

We select generative baselines that share our problem setting (joint $p(a, u)$ prior, sparse-observation guidance). Together, they also cover the design choices we want to compare against: diffusion vs. MeanFlow backbones, physics supervision applied during training vs. only at inference, and iterative guided sampling vs. one-step distillation with test-time optimization.

Classical neural operators (PINO, DeepONet, FNO) and physics-informed neural networks (PINNs) target dense-observation regimes and perform poorly in this sparse-observation setting (Huang et al., 2024). We exclude them from the main comparison to avoid an unfair setting. We compare against the following generative baselines:

- **DiffusionPDE**[2] (Huang et al., 2024): the closest prior work, which combines a diffusion prior with observational and PDE-residual guidance at inference.
- **PIDM**[3] (Bastek et al., 2025): physics-informed diffusion with a PDE-residual training loss. We retrain PIDM on our datasets and apply the DiffusionPDE guidance procedure at inference, using its physics-trained denoiser as the score model.
- **PIDDM** (Zhang & Zou, 2025): physics-informed distillation of a pretrained Flow Matching teacher into a one-step generator, followed by test-time optimization of the latent under observation and PDE-residual losses. We re-implement it on our benchmarks; the protocol is described in App. I.
- **FmPDE**: the pretrained Flow Matching model that initializes PIMF-FMinit and serves as the PIDDM teacher, run with the same DiffusionPDE guidance procedure, with the constraints eval-

[2]We re-run their method with their checkpoints and parameters from the repository.
[3]We use DiffusionPDE's diffusion models as initialization.

uated on its one-step Euler endpoint estimate. It isolates the contribution of the MeanFlow parameterization and of physics-informed training from that of the flow backbone.

We evaluate two variants of our method: **PIMF**, trained entirely from scratch following the two-stage procedure of Section 4.1; and **PIMF-FMinit**, which only differs from the previous one in the initialization from a pretrained Flow Matching model. PIMF-FMinit starts from the same pretrained Flow Matching model that serves as the PIDDM teacher. We also provide a theoretical complexity analysis of the baselines in App. F.

5.3 Reconstruction Accuracy

Table 1 reports relative error across all reconstruction tasks; all means and standard deviations are also collected with both reconstructed channels in Table 6. The main findings are as follows.

Table 1: Target-field relative error in percent (mean $\pm$ standard deviation over 1000 test instances) with $N = 2000$ and 500 observations (five sensor columns for Burgers). Forward rows evaluate u and inverse rows evaluate a. FmPDE is the guided Flow Matching teacher of PIMF-FMinit (Section 5.2). Bold marks the lowest mean and underline the second-lowest.

PDE task	DiffusionPDE	PIDM	PIDDM	FmPDE	PIMF	PIMF-FMinit
Darcy (Fwd u)	5.97 ± 4.40	5.17 ± 4.20	8.06 ± 3.94	$\mathbf{3.99 \pm 2.65}$	6.09 ± 2.30	$\underline{4.18 \pm 2.06}$
Darcy (Inv a)	7.88 ± 4.50	$\mathbf{6.47 \pm 3.70}$	31.51 ± 10.44	$\underline{6.60 \pm 4.17}$	8.17 ± 4.90	9.33 ± 6.10
Poisson (Fwd u)	4.98 ± 2.50	$\underline{1.97 \pm 1.00}$	2.71 ± 1.57	2.57 ± 1.25	3.33 ± 1.80	$\mathbf{1.66 \pm 0.84}$
Poisson (Inv a)	21.20 ± 6.20	19.40 ± 5.90	21.85 ± 6.38	15.68 ± 4.40	$\underline{14.76 \pm 4.30}$	$\mathbf{13.99 \pm 4.21}$
Helmholtz (Fwd u)	10.95 ± 5.60	$\underline{1.95 \pm 1.00}$	2.70 ± 1.37	2.29 ± 1.20	2.32 ± 1.10	$\mathbf{1.70 \pm 0.92}$
Helmholtz (Inv a)	18.98 ± 5.20	$\mathbf{14.06 \pm 4.00}$	21.34 ± 6.07	15.77 ± 4.39	15.35 ± 5.20	$\underline{14.51 \pm 4.25}$
NSNB (Fwd u)	3.78 ± 0.70	$\underline{3.06 \pm 0.70}$	3.49 ± 0.80	3.95 ± 0.99	3.28 ± 0.80	$\mathbf{2.95 \pm 0.73}$
NSNB (Inv a)	9.64 ± 1.80	8.96 ± 1.80	12.61 ± 2.64	8.97 ± 1.81	$\underline{8.67 \pm 1.70}$	$\mathbf{8.27 \pm 1.63}$
Burgers	10.40 ± 12.80	8.97 ± 12.40	$\underline{4.84 \pm 7.25}$	21.51 ± 16.04	6.06 ± 6.00	$\mathbf{2.36 \pm 1.55}$

(1) PIMF-FMinit improves most of the target-field reconstructions. PIMF-FMinit achieves the lowest relative error on six of the nine tasks. The exceptions are the two Darcy tasks, where FmPDE (forward) and PIDM (inverse) are more accurate, and Helmholtz inverse, where PIDM is marginally better. We explain the Darcy behavior in the limitations (Section 7).

(2) Guided MeanFlow sampling outperforms distillation with test-time optimization. PIDDM is distilled from the same Flow Matching checkpoint that initializes PIMF-FMinit, yet PIMF-FMinit achieves lower error on every task.

(3) The gains are not explained by the flow backbone alone. FmPDE shares its weights with the initialization of PIMF-FMinit, yet PIMF-FMinit achieves lower error on seven of the nine tasks; FmPDE is more accurate on both Darcy tasks.

We provide additional ablations in App. E, including the residual norm and late-guidance schedule.

5.4 The Role of Physics

To isolate the two different uses of physics information, we perform the following ablation study. We cross the PIMF-FMinit checkpoints after standard Stage 1 training and physics-informed Stage 2 training with observation-only and observation-and-PDE guidance at inference. We omit NSNB because it uses a proxy residual, and Burgers because it is single-channel.

Table 2 separates the effects of training-time and inference-time physics. Observation-only inference already performs strongly on Helmholtz and Poisson. Adding PDE guidance improves the elliptic forward tasks but degrades Helmholtz and Poisson inverse reconstruction, where observation-only guidance is best. We still keep DiffusionPDE's schedule in Table 1 for comparability. Stage 2 improves the forward tasks, is neutral on Helmholtz inverse and on Poisson inverse under observation-only guidance, reduces the Poisson inverse error under observation-plus-PDE guidance, and is detrimental on Darcy inverse. Thus, the consistent reconstruction gains are already present with

observation-guided MeanFlow inference, while the physics components provide task-dependent refinements.

Table 2: Component analysis: target relative error in percent (mean $\pm$ standard deviation over 100 instances). Columns compare observation-only and observation-plus-PDE guidance for checkpoints after Stage 1 and Stage 2. Bold marks the lowest mean within each task.

Task	Stage 1, observation	Stage 1, observation + PDE	Stage 2, observation	Stage 2, observation + PDE
Helmholtz forward (u)	2.30 ± 0.97	1.72 ± 0.81	1.92 ± 0.89	$\mathbf{1.70 \pm 0.83}$
Poisson forward (u)	3.48 ± 1.52	2.25 ± 0.98	1.92 ± 0.94	$\mathbf{1.77 \pm 0.86}$
Darcy forward (u)	4.81 ± 4.26	4.39 ± 4.32	4.16 ± 1.87	$\mathbf{4.07 \pm 1.79}$
Helmholtz inverse (a)	$\mathbf{12.41 \pm 3.32}$	14.55 ± 3.92	12.50 ± 3.45	14.56 ± 4.06
Poisson inverse (a)	13.33 ± 4.29	15.31 ± 5.02	$\mathbf{13.27 \pm 4.56}$	14.40 ± 4.86
Darcy inverse (a)	$\mathbf{8.06 \pm 5.27}$	8.07 ± 5.27	8.78 ± 5.98	8.78 ± 5.98

5.5 Inference Efficiency

Inference efficiency measures reconstruction quality as a function of the sampling budget N. To test whether PIMF's advantage in Table 1 survives smaller budgets, we evaluate every method at reduced budgets $\{50, 100, ..., 1000\}$, keeping the full-budget hyperparameters (App. C) and reporting the mean relative error over the test set. PIMF's clearest gains occur on Helmholtz and Poisson, with a weaker advantage on Darcy (Figure 2; full results in Figure 5). PIDDM is better only at 50 steps.

The trajectories in Figure 5 also show when each sampler begins to match the constraints: PIMF reduces observation error earlier on Helmholtz, Poisson, and NSNB forward recovery, whereas Darcy and Burgers show weaker or no separation (enlarged Helmholtz view in Figure 2). FmPDE, whose iteration is the cheapest (Table 9), converges faster than the diffusion baselines but more slowly than PIMF at small budgets.

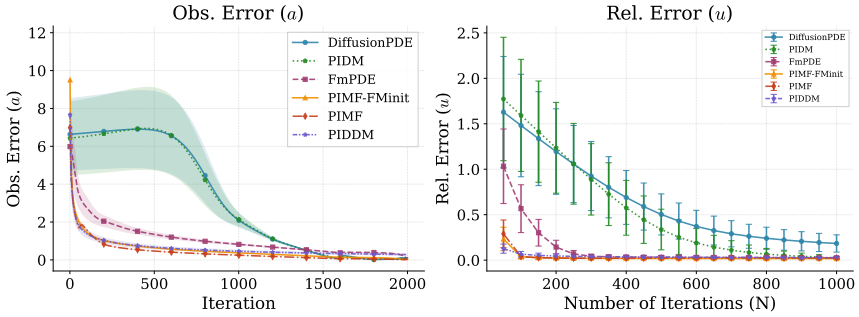

Figure 2: **Helmholtz forward reconstruction. Left:** guidance trajectories at $N = 2000$. The coefficient a is sparsely observed and the solution u is unobserved. PIMF (on par with PIDDM) reduces the observation errors earlier in the trajectory. **Right:** relative error of the target field u versus sampling budget N on the same task.

5.6 Novel Solution Discovery

Here, novel refers to candidate reconstructions that are distinct under the same sparse observations, rather than to a capability unique to PIMF. The reduced-iteration regime makes it practical to explore the ambiguity of sparse inverse problems through independent initial states. All multi-start and ensemble experiments in this section and in App. H use the PIMF-FMinit checkpoint. Each run uses the same measurements but a different initial state, producing an empirical collection of reconstructions under a fixed observation condition. Figure 3 shows the Helmholtz example: the candidates retain the sparse constraints while differing in unobserved regions and contain visibly distinct solution fields. The large variability stems from having only 10 observations (0.06% of points). App. H reports cluster representatives, constraint checks, mode identification, and the relation between spatial disagreement and reconstruction error. These candidates support solution discovery and ambiguity analysis, but are not interpreted as calibrated posterior modes.

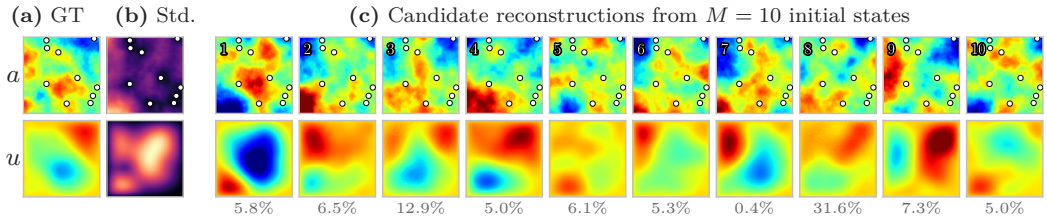

Figure 3: a) Ground truth samples with only 10 observations on a (white). b) Ensemble dispersion (pixel-wise standard deviation across the 10 PIMF-FMinit candidates). c) 10 different new solutions where all but one closely match the observations and have low residuals.

6 Related Work

Guidance for inverse problems. Diffusion Posterior Sampling (Chung et al., 2023) established inference-time guidance through a clean-state estimate, and DiffusionPDE (Huang et al., 2024) added PDE-residual guidance for joint coefficient–solution priors. Subsequent work improves diffusion guidance through additional moments, variational or pseudoinverse corrections, and frequency-dependent operators (Boys et al., 2024; Mardani et al., 2024; Song et al., 2023a; Thaker et al., 2025). Few-step guidance has also been studied with consistency models (Zhao et al., 2024), while function-space diffusion methods target discretization-independent PDE inversion (Yao et al., 2026; Lin et al., 2026). PIMF instead uses the finite-horizon projection of a MeanFlow prior within an iterative sparse-reconstruction solver. Flow Matching-based methods like Extrapolation-Correction-Interpolation (Cheng et al., 2025) and Physics-Constrained Flow Matching (Utkarsh et al., 2025) continuously project intermediate states onto a constraint manifold during sampling, but require the coefficient or solution to be fully specified (e.g., the entire initial condition) rather than reconstructed from sparse measurements, and are therefore not directly comparable to our setting.

Physics-informed generative models. Physics-Informed Diffusion Models (PIDM) incorporate PDE and boundary residuals into diffusion training (Bastek et al., 2025), and Physics-Informed Distillation of Diffusion Models (PIDDM) combines physics-informed distillation with a few-step generator (Zhang & Zou, 2025). Physics-Based Flow Matching adds residuals to Flow-Matching training (Baldan et al., 2025); related flow methods fine-tune or project trajectories toward physical constraints (Utkarsh et al., 2025; Tauberschmidt et al., 2026). PIMF combines staged residual fine-tuning with observation and residual guidance evaluated on a MeanFlow projection.

7 Limitations

Training overhead. Stage 2 adds one gradient-free network evaluation at $(0, r)$ per gated sample and backpropagates through the PDE residual, making its iterations moderately more expensive than plain MeanFlow training (App. A).

Underperformance on Darcy inverse. Darcy Flow's binary permeability field a creates sharp interfaces, and small errors there, which noisy samples cannot avoid, dominate the training residual $\|\nabla \cdot (a\nabla u)\|_\infty$. Several baselines therefore beat both PIMF variants on Darcy, even ones trained without a PDE loss such as DiffusionPDE, most clearly on inverse reconstruction.

8 Conclusion

We present PIMF, an alternative to diffusion-based inverse solvers that uses the MeanFlow projection for guidance. Across the evaluated benchmarks, PIMF improves reconstruction accuracy on most tasks and converges faster than the diffusion-based solvers across several PDE families. Its efficiency on par with PIDDM also supports the discovery of distinct observation-consistent candidate solutions.

AI USE STATEMENT

We used generative AI tools for three purposes.

1. Since the authors of PIDDM (Zhang & Zou, 2025) have not released code, we used an agentic coding assistant to re-implement it from the paper (and we did many rounds of implementation and testing, and took the best-performing implementation) and tuned it more extensively than the other baselines on held-out instances, following the protocol in Section I. Most of the remaining code was written without agentic tools (but with chat-based LLMs).
2. We also asked LLMs for feedback on our experimental design to check for missing baselines or ablations.
3. The main PIMF code was written by the authors, with LLMs assisting in debugging.
4. We used LLMs to help drafting and polishing the paper.

For the remaining required and recommended points, we did not use generative AI to formulate mathematical claims or proofs, propose hypotheses, generate or clean datasets (as we strictly use the datasets from Huang et al. (2024)), or interpret results, and translation and qualitative analysis do not apply to this work.

REPRODUCIBILITY STATEMENT

All components needed to reproduce our results are described in the paper. The two-stage training procedure and the physics-informed objective are defined in Section 4.1 (Equation (7)), with the full training loop given in Algorithm 2 and the guided inference procedure in Algorithm 1. The PDE families, observation settings, test splits, and evaluation metric are specified in Section 5.1. The governing equations, guidance functions, and physical parameters are given in full in Section B. We reuse the publicly available datasets of Huang et al. (2024) without modification. Section A reports the network architecture and every training hyperparameter for both PIMF variants, including the stage split, physics weight, residual norm, gating threshold, optimizer settings, batch size, training duration, hardware, and the checkpoint that PIMF-FMinit is initialized from. Section C lists the guidance weights and schedule used for every method and dataset, the procedure used to tune them, and the full reconstruction protocol. Because the authors of PIDDM (Zhang & Zou, 2025) have not released code, Section I documents our re-implementation, including the distillation setup, hyperparameter grids, and the tuning instances. Upon acceptance, the code for training, guided inference, and evaluation, including our baseline re-implementations, will be released on GitHub, and the trained model checkpoints and reconstructed samples will be made available on Zenodo.

ETHICS STATEMENT

All authors adhere to the ICLR Code of Ethics. This work involves no human subjects or personal data. All experiments use the public synthetic PDE datasets of Huang et al. (2024). Since our results are on synthetic benchmarks, PIMF should not inform safety-critical decisions without validation on real data.

REFERENCES

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Giacomo Baldan, Qiang Liu, Alberto Guardone, and Nils Thuerey. Flow matching meets pdes: A unified framework for physics-constrained generation. *arXiv preprint arXiv:2506.08604*, 2025.

Jan-Hendrik Bastek, WaiChing Sun, and Dennis Kochmann. Physics-informed diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=tpYeermigp`.

Martin Benning and Martin Burger. Modern regularization methods for inverse problems. *Acta numerica*, 27:1–111, 2018.

Martin Benning, Lynn Gladden, Daniel Holland, Carola-Bibiane Schönlieb, and Tuomo Valkonen. Phase reconstruction from velocity-encoded mri measurements–a survey of sparsity-promoting variational approaches. *Journal of Magnetic Resonance*, 238:26–43, 2014.

Claus Borgnakke. *Fundamentals of thermodynamics*. John Wiley & Sons, 2025.

Benjamin Boys, Mark Girolami, Jakiw Pidstrigach, Sebastian Reich, Alan Mosca, and Omer Deniz Akyildiz. Tweedie moment projected diffusions for inverse problems. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=4unJi0qrTE. Featured Certification.

Che-Chia Chang, Chen-Yang Dai, Te-Sheng Lin, Ming-Chih Lai, and Chieh-Hsin Lai. Consistency training with physical constraints. In *Frontiers in Probabilistic Inference: Learning meets Sampling*, 2025. URL https://openreview.net/forum?id=k8vbqHaCnQ.

Yuqing Chen and Erdinc Saygin. Seismic inversion by hybrid machine learning. *Journal of Geophysical Research: Solid Earth*, 126(9):e2020JB021589, 2021.

Chaoran Cheng, Boran Han, Danielle C Maddix, Abdul Fatir Ansari, Andrew Stuart, Michael W Mahoney, and Bernie Wang. Gradient-free generation for hard-constrained systems. In *The Thirteenth International Conference on Learning Representations*, 2025.

Hyungjin Chung, Jeongsol Kim, Michael T McCann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Giannis Daras, Hyungjin Chung, Chieh-Hsin Lai, Yuki Mitsufuji, Jong Chul Ye, Peyman Milanfar, Alexandros G Dimakis, and Mauricio Delbracio. A survey on diffusion models for inverse problems. *arXiv preprint arXiv:2410.00083*, 2024.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Michael Eckert. *The dawn of fluid dynamics: a discipline between science and technology*. John Wiley & Sons, 2007.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11, 2018.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=uWj4s7rMnR.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/glorot10a.html.

Yi Guo, Wei Wang, Zhihang Yuan, Rong Cao, Kuan Chen, Zhengyang Chen, Yuanyuan Huo, Yang Zhang, Yuping Wang, Shouda Liu, et al. Splitmeanflow: Interval splitting consistency in few-step generative modeling. *arXiv preprint arXiv:2507.16884*, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. Diffusionpde: Generative pde-solving under partial observation. *Advances in Neural Information Processing Systems*, 37: 130291–130323, 2024.

Eric Jonas. Deep imitation learning for molecular inverse problems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/b0bef4c9a6e50d43880191492d4fc827-Paper.pdf.

Eugenia Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 26565–26577. Curran Associates, Inc., 2022.

Rihuan Ke and Carola-Bibiane Schönlieb. Unsupervised image restoration using partially linear denoisers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5796–5812, 2021.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In *The Twelfth International Conference on Learning Representations*, 2024.

Chieh-Hsin Lai, Yang Song, Dongjun Kim, Yuki Mitsufuji, and Stefano Ermon. The principles of diffusion models. *arXiv preprint arXiv:2510.21890*, 2025.

Thomas YL Lin, Jiachen Yao, Lufang Chiang, Julius Berner, and Anima Anandkumar. Decoupled diffusion sampling for inverse problems on function spaces. *arXiv preprint arXiv:2601.23280*, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Morteza Mardani, Jiaming Song, Jan Kautz, and Arash Vahdat. A variational perspective on solving inverse problems with diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

Carola-Bibiane Schönlieb and Zakhar Shumaylov. Data-driven approaches to inverse problems, 2025. URL https://arxiv.org/abs/2506.11732.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=9_gsMA8MRKQ.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations*, 2024.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vaRCHVj0uGI.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pp. 32211–32252. PMLR, 2023b.

Jan Tauberschmidt, Sophie Fellenz, Sebastian Vollmer, and Andrew Duncan. Physics-constrained fine-tuning of flow-matching models for generation and inverse problems. In *International Conference on Learning Representations*, volume 2026, pp. 143992–144037, 2026.

Darshan Thaker, Abhishek Goyal, and René Vidal. Frequency-guided posterior sampling for diffusion-based image restoration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12873–12882, 2025.

Utkarsh Utkarsh, Pengfei Cai, Alan Edelman, Rafael Gomez-Bombarelli, and Christopher Vincent Rackauckas. Physics-constrained flow matching: Sampling generative models with hard constraints. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=cf4etwjY7n.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Chuwei Wang, Shanda Li, Di He, and Liwei Wang. Is $l2$ physics informed loss always suitable for training physics informed neural network? *Advances in Neural Information Processing Systems*, 35:8278–8290, 2022.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

Jiachen Yao, Abbas Mammadov, Julius Berner, Gavin Kerrigan, Jong Chul Ye, Kamyar Azizzadenesheli, and Animashree Anandkumar. Guided diffusion sampling on function spaces with applications to pdes. *Advances in Neural Information Processing Systems*, 38:127057–127094, 2026.

Yi Zhang and Difan Zou. Physics-informed distillation of diffusion models for pde-constrained generation. *arXiv preprint arXiv:2505.22391*, 2025.

Jiankun Zhao, Bowen Song, and Liyue Shen. Cosign: Few-step guidance of consistency model to solve general inverse problems. In *European Conference on Computer Vision*, pp. 108–126. Springer, 2024.

Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. In *Forty-second International Conference on Machine Learning*, 2025.

APPENDIX CONTENTS

A Architecture and Training Details **15**
A.1 Network Architecture 15
A.2 Training Configuration 15

B PDE Families **16**

C Inference Hyperparameters and Evaluation Protocol **17**
C.1 Guidance Weights 17
C.2 Reconstruction Protocol and Statistics 17

D Full Reconstruction Results **18**

E Additional Physics Ablations **18**
E.1 Sensitivity to the Late-Guidance Schedule 18
E.2 Residual Loss 19
E.3 Matched-Duration Control Without Physics 19

F Runtime and Complexity **20**

G Additional Trajectory and Sampling-Budget Figures **21**

H Additional Multi-Start Analysis **23**
H.1 Novel-Solution Diagnostics 23
H.2 Ensemble Dispersion Metrics 24
H.3 Spatial Disagreement and Reconstruction Error 25

I PIDDM Baseline: Distillation and Tuning Protocol **26**

J PyTorch-Style Pseudocode **27**

K Qualitative Reconstructions **29**

A Architecture and Training Details

A.1 Network Architecture

For a direct comparison with DiffusionPDE (Huang et al., 2024), we retain its SongUNet backbone and EDM preconditioning as closely as the MeanFlow parameterization permits (Song et al., 2021; Karras et al., 2022). The coefficient and solution fields are concatenated along the channel dimension. MeanFlow requires two time arguments (r, t) rather than one: we embed them separately with sinusoidal positional embeddings, concatenate the embeddings, and project the result back to dimension 512 with a linear layer.

The U-Net has base width 128 and feature widths $[128, 256, 256]$, with four residual blocks at each resolution and self-attention at 16×16. Encoder–decoder skip connections are scaled by $\sqrt{0.5}$. We use Group Normalization with $\epsilon = 10^{-6}$, SiLU activations, dropout 0.13, Xavier-uniform initialization, and FP32 training (Vaswani et al., 2017; Wu & He, 2018; Elfwing et al., 2018; Glorot & Bengio, 2010). There is no label dropout because the prior is unconditional.

A.2 Training Configuration

Table 3 collects the numerical values used in the reported training runs. Each PDE dataset contains 50,000 training pairs on a 128×128 grid. The reported models were trained on four NVIDIA A40 or A100 GPUs. PIMF-FMinit starts from the same pretrained Flow-Matching checkpoint that serves as the PIDDM teacher (App. I); PIMF starts from a random Xavier initialization.

Table 3: Architecture and training hyperparameters used by both PIMF variants. The optimization horizon is divided equally between the two stages; PIMF-FMinit additionally incurs the cost of its Flow-Matching pretraining.

Setting	Value	Setting	Value
Training pairs	50,000	Spatial resolution	128×128
Base width	128	Feature widths	$[128, 256, 256]$
Residual blocks/level	4	Attention resolution	16×16
Time-embedding size	512	Dropout	0.13
Precision	FP32	Training devices	4 GPUs
Flow ratio q_{FM}	0.6	Stage split	50% / 50%
Physics weight λ_{pde}	10^{-3}	Training residual norm	L^{∞}
Physics gate $r_{\max}$	0.15	Endpoint evaluation	stop-gradient
Optimizer	Adam	Learning rate	10^{-4}
Adam (β_1, β_2)	$(0.9, 0.999)$	Batch size	64 (Stage 1: 32–64)
EMA half-life	50 kimg	Total duration	20,000 kimg
Reported checkpoint	Final EMA weights at 20,000 kimg. Same budget as DiffusionPDE.		

Both variants use a linear probability path and the SplitMeanFlow objective $\mathcal{L}_{\mathrm{SMF}}$ of Section 3.2 (Guo et al., 2025). Stage 1 occupies the first half of optimization. During the second half, Stage 2 continues MeanFlow training and adds the endpoint residual from Equation (7) on interval-splitting samples with $r < r_{\max}$. The endpoint evaluation at $(0, r)$ is computed under a stop-gradient, so Stage 2 adds a network evaluation without storing its activations; gradients reach θ only through the first hop, which reuses the interval-splitting branch's prediction. Two implementation details differ from the single-sample notation of Equation (7) and Algorithm 2. First, the training branch (flow matching with $r = t$, or interval splitting) is drawn once per minibatch and shared by all of its samples, while t, r, and γ are drawn per sample. Second, the L^{∞} norm of the Stage 2 residual is taken jointly over all gated samples of the minibatch, i.e., it is the batch maximum rather than a per-sample maximum; the resulting scalar is added to the loss of every gated sample. Section J gives the corresponding batched pseudocode.

Algorithm 2 Two-stage Physics-Informed MeanFlow training

Require: Network m_θ; data p_{data}; total updates S; $q_{\text{FM}} = 0.6$; $\lambda_{\text{pde}} = 10^{-3}$; $r_{\max} = 0.15$

for $k = 0, \dots, S-1$ **do**
  Sample $x_0 \sim p_{\text{data}}$, $x_1 \sim \mathcal{N}(0, I)$, and $t \sim \mathcal{U}[0,1]$
  $x_t \leftarrow (1-t)x_0 + tx_1$
  if $b \sim \text{Bernoulli}(q_{\text{FM}})$ (one draw per minibatch, shared by all samples) **then**
    $r \leftarrow t$; $m_{\text{pred}} \leftarrow m_\theta(x_t, t, t)$; $m_{\text{tgt}} \leftarrow x_1 - x_0$
  else
    Sample $r \sim \mathcal{U}[0,t]$, $\gamma \sim \mathcal{U}[0,1]$; $s \leftarrow (1-\gamma)t + \gamma r$
    $m_{\text{pred}} \leftarrow m_\theta(x_t, r, t)$
    $u_2 \leftarrow \text{sg}(m_\theta(x_t, s, t))$; $x_s \leftarrow x_t - (t-s)u_2$
    $u_1 \leftarrow \text{sg}(m_\theta(x_s, r, s))$
    $m_{\text{tgt}} \leftarrow (1-\gamma)u_1 + \gamma u_2$
  end if
  $\mathcal{L} \leftarrow \text{mean}\left[(m_{\text{pred}} - \text{sg}(m_{\text{tgt}}))^2\right]$ (mean over all entries)
  if $k \geq S/2$, the splitting branch was taken, and $r < r_{\max}$ **then**
    $x_r \leftarrow x_t - (t-r)m_{\text{pred}}$
    $\hat{x}_0 \leftarrow x_r - r\,\text{sg}(m_\theta(x_r, 0, r))$
    $\mathcal{L} \leftarrow \mathcal{L} + \lambda_{\text{pde}} \|\mathcal{F}(\hat{x}_0)\|_\infty$
  end if
  Update θ using $\nabla_\theta \mathcal{L}$
end for
return Final Stage 2 checkpoint

B PDE FAMILIES

We reserve t for generative time and τ for physical time. All arrays have resolution 128×128. For the paired problems, a denotes the coefficient or earlier state and u the solution or later state.

Table 4: Benchmark families and residuals. The NSNB entry follows the simplified proxy used by DiffusionPDE, not a full Navier–Stokes residual.

Family	Domain/representation	Physical parameter	Residual used by PIMF
Darcy	$(0,1)^2$	$q = 1$	$\nabla\cdot(a\nabla u) + q$
Poisson	$(0,1)^2$	$\kappa = 0$	$\Delta u + \kappa^2 u - a$
Helmholtz	$(0,1)^2$	$\kappa = 1$	$\Delta u + \kappa^2 u - a$
NSNB	two vorticity snapshots	$\nu = 10^{-3}$	$\nabla\cdot\omega$ (proxy)
Burgers	space–time field	$\nu = 10^{-2}$	$\partial_\tau u + \partial_c(u^2/2) - \nu\partial_{cc}u$

Darcy flow. On $\Omega = (0,1)^2$ with homogeneous Dirichlet conditions,

$$-\nabla\cdot(a(c)\nabla u(c)) = q(c), \qquad q(c) = 1. \tag{8}$$

The binary permeability contains sharp interfaces, so differential residuals are particularly sensitive to the grid discretization at coefficient boundaries.

Poisson and Helmholtz. We use

$$\Delta u(c) + \kappa^2 u(c) = a(c), \tag{9}$$

with $\kappa = 0$ for Poisson and $\kappa = 1$ for Helmholtz, both with homogeneous Dirichlet conditions. In forward recovery, applying the inverse elliptic operator smooths error. In inverse recovery, applying the differential operator to an imperfect u can amplify its high-frequency error.

Burgers. The complete array represents a one-dimensional space–time trajectory on a periodic spatial domain:

$$\partial_\tau u + \partial_c(u^2/2) - \nu\partial_{cc}u = 0, \qquad \nu = 10^{-2}. \tag{10}$$

Unbounded Navier–Stokes. The data contain initial and later scalar-vorticity snapshots governed by

$$\partial_\tau \omega + v \cdot \nabla \omega = \nu \Delta \omega + q, \qquad \nabla \cdot v = 0. \tag{11}$$

Two endpoint snapshots do not determine the complete temporal residual without a simulator or trajectory model, and incompressibility applies to velocity v, not scalar vorticity ω. Following DiffusionPDE, the proxy is written as

$$f_{\mathrm{NSNB}}(\omega) = \nabla \cdot \omega(c, \tau). \tag{12}$$

In our endpoint setting, it is evaluated on the predicted later snapshot $u = \omega(\cdot, T)$. We retain the benchmark's notation for reproducibility; this proxy is neither the velocity incompressibility residual $\nabla \cdot v$ nor the full dynamical residual. It is already at the numerical floor of the discretization and supplies essentially no guidance. We therefore retain NSNB as a reconstruction benchmark but exclude it from every physical-consistency conclusion.

C Inference Hyperparameters and Evaluation Protocol

C.1 Guidance Weights

Table 5 lists the guidance weights used by PIMF and PIMF-FMinit. DiffusionPDE is run with the weights released by its authors; PIDM and FmPDE weights were tuned with the same procedure as ours: a grid search on a single instance per dataset (offset 1 of the benchmark test split, which is therefore not held out), after which the selected weights were applied unchanged to all 1000 test instances. PIDDM uses the protocol of App. I.

We adopt the DiffusionPDE schedule unchanged for all methods, with $\delta_{\mathrm{pde}} = 0.8$ and $\alpha_{\mathrm{obs}} = 0.1$: observation guidance is active alone before iteration $\lfloor \delta_{\mathrm{pde}} N \rfloor$, after which PDE guidance is activated and the observation contribution is rescaled by α_{obs}. Losses are differentiated through the estimator used by each method, and the updated state is detached between iterations.

Table 5: Guidance weights. Slots 1 and 2 contain a and u for paired problems; Burgers has one field. For NSNB, ζ_{pde} is given as forward / inverse.

	Poisson	Darcy	Helmholtz	NSNB	Burgers
$\zeta_{\mathrm{obs}}^{(1)}$	0.2	0.4	0.2	0.2	100
$\zeta_{\mathrm{obs}}^{(2)}$	80	200	60	0.8	–
ζ_{pde}	10^{-1}	10^{-1}	10^{-1}	10 / 1	30

C.2 Reconstruction Protocol and Statistics

Following DiffusionPDE, every main reconstruction uses 500 point observations from only one channel, equivalent to 3.05% of a 128×128 field (Burgers instead observes five sensor columns of the space–time field, i.e., 640 points). Forward tasks observe a and evaluate the recovered u; inverse tasks observe u and evaluate the recovered a. Burgers reconstructs the complete space–time trajectory from partial values of the same field. The main comparison uses $N = 2000$ guided iterations.

For n test instances, we report the samplewise spatial relative error

$$\mathrm{Rel}(\hat{x}, x) = \frac{100}{n} \sum_{j=1}^{n} \frac{\|\hat{x}^{(j)} - x^{(j)}\|_2}{\|x^{(j)}\|_2}. \tag{13}$$

Values in tables are percentages. The $\pm$ entries in reconstruction tables are standard deviations across test instances, whereas trajectory and sampling-budget bands show ± 1 standard error. The main table uses 1000 instances per task; trajectory curves, budget curves, and the checkpoint–guidance comparison use 100.

D Full Reconstruction Results

The main paper reports the channel that defines each forward or inverse task. For completeness, Table 6 reports both reconstructed channels.

Table 6: Relative error in percent (mean ± standard deviation over 1000 test instances) for both reconstructed channels. Bold marks the lowest mean and underline the second-lowest.

PDE	Task/channel	DiffusionPDE	PIDM	PIDDM	FmPDE	PIMF	PIMF-FMinit
Darcy	Fwd a	$\mathbf{4.37 \pm 1.90}$	4.65 ± 2.20	24.44 ± 5.98	4.85 ± 2.50	7.84 ± 2.80	5.64 ± 2.27
	Fwd u	5.97 ± 4.40	5.17 ± 4.20	8.06 ± 3.94	$\mathbf{3.99 \pm 2.65}$	6.09 ± 2.30	4.18 ± 2.06
	Inv a	7.88 ± 4.50	$\mathbf{6.47 \pm 3.70}$	31.51 ± 10.44	6.60 ± 4.17	8.17 ± 4.90	9.33 ± 6.10
	Inv u	7.07 ± 3.00	10.52 ± 3.50	1.78 ± 0.70	1.54 ± 0.90	1.16 ± 0.40	$\mathbf{1.06 \pm 0.65}$
Poisson	Fwd a	10.31 ± 3.10	10.23 ± 3.10	10.62 ± 3.08	11.29 ± 3.40	9.93 ± 3.10	$\mathbf{9.88 \pm 2.98}$
	Fwd u	4.98 ± 2.50	1.97 ± 1.00	2.71 ± 1.57	2.57 ± 1.25	3.33 ± 1.80	$\mathbf{1.66 \pm 0.84}$
	Inv a	21.20 ± 6.20	19.40 ± 5.90	21.85 ± 6.38	15.68 ± 4.40	14.76 ± 4.30	$\mathbf{13.99 \pm 4.21}$
	Inv u	0.517 ± 0.20	0.498 ± 0.30	2.31 ± 2.22	0.676 ± 0.30	$\mathbf{0.490 \pm 0.20}$	0.863 ± 0.53
Helmholtz	Fwd a	10.48 ± 3.00	10.40 ± 3.00	10.93 ± 3.01	11.24 ± 3.30	10.21 ± 3.00	$\mathbf{10.20 \pm 3.00}$
	Fwd u	10.95 ± 5.60	1.95 ± 1.00	2.70 ± 1.37	2.29 ± 1.20	2.32 ± 1.10	$\mathbf{1.70 \pm 0.92}$
	Inv a	18.98 ± 5.20	$\mathbf{14.06 \pm 4.00}$	21.34 ± 6.07	15.77 ± 4.39	15.35 ± 5.20	14.51 ± 4.25
	Inv u	0.949 ± 0.40	$\mathbf{0.394 \pm 0.20}$	2.30 ± 2.18	2.72 ± 0.50	2.38 ± 0.20	0.576 ± 0.25
NSNB	Fwd a	9.47 ± 2.00	9.38 ± 2.00	10.19 ± 2.05	11.30 ± 2.30	9.56 ± 2.00	$\mathbf{9.25 \pm 1.96}$
	Fwd u	3.78 ± 0.70	3.06 ± 0.70	3.49 ± 0.80	3.95 ± 0.99	3.28 ± 0.80	$\mathbf{2.95 \pm 0.73}$
	Inv a	9.64 ± 1.80	8.96 ± 1.80	12.61 ± 2.64	8.97 ± 1.81	8.67 ± 1.70	$\mathbf{8.27 \pm 1.63}$
	Inv u	0.690 ± 0.20	$\mathbf{0.622 \pm 0.20}$	2.11 ± 1.11	1.621 ± 0.40	4.462 ± 1.00	3.833 ± 1.04
Burgers	Full trajectory	10.40 ± 12.80	8.97 ± 12.40	4.84 ± 7.25	21.51 ± 16.04	6.06 ± 6.00	$\mathbf{2.36 \pm 1.55}$

Boldface in the task rows identifies the lowest reported mean among the evaluated methods; observed-channel rows are included for context and are not used to count task wins. On NSNB inverse, the observed channel u is recovered less accurately by both PIMF variants than by the other methods. The PIMF observation weight for this task (Table 5) was selected for the target field a, so the unobserved part of u is less tightly constrained, even though the target-field error is the lowest among all methods. The full standard deviations describe variation across test examples.

E Additional Physics Ablations

E.1 Sensitivity to the Late-Guidance Schedule

The main experiments retain DiffusionPDE's 80/20 schedule for every method rather than tune the split for PIMF. This is done intentionally to keep the results comparable (also in terms of gradient calls and cost). Nevertheless, we also show the results for other schedules. We vary the observation-only/observation-plus-PDE allocation for the PIMF-FMinit Stage 2 checkpoint over 2000 iterations and the first 100 test instances.

Table 7: Target relative error in percent (mean ± standard deviation) as the observation-only/PDE-refinement split changes.

Task	60/40	70/30	80/20 (default)	90/10
Helmholtz forward (u)	1.74 ± 0.87	1.71 ± 0.84	1.70 ± 0.83	1.72 ± 0.84
Poisson forward (u)	1.80 ± 0.82	1.78 ± 0.84	1.77 ± 0.86	1.77 ± 0.88
Helmholtz inverse (a)	16.08 ± 4.57	15.35 ± 4.31	14.56 ± 4.06	13.63 ± 3.77
Poisson inverse (a)	15.23 ± 5.09	14.82 ± 4.96	14.40 ± 4.86	13.98 ± 4.82

Forward errors are nearly flat across this range. Inverse coefficient error improves monotonically as the PDE-guided fraction decreases, consistent with the task-dependent effect of inference-time physics in Section 5.4.

E.2 Residual Loss

We use the L^∞ norm of the PDE residual as the physics-informed training loss in PIMF training, following Wang et al. (2022); Chang et al. (2025). They show that the L^2 residual can misrepresent the true solution error, because the error may concentrate at a few high-gradient points whose contribution L^2 averaging washes out. By penalizing the worst-case points, the L^∞ norm (or in general, an L^p norm with high p) can bound the solution error under the stability assumptions stated in those works, and empirically enhances generalization in physics-informed training. Our experiments confirmed this: using an L^2 residual loss for retraining the Darcy model led to noticeably worse PDE residuals on generated samples, as seen in Figure 4.

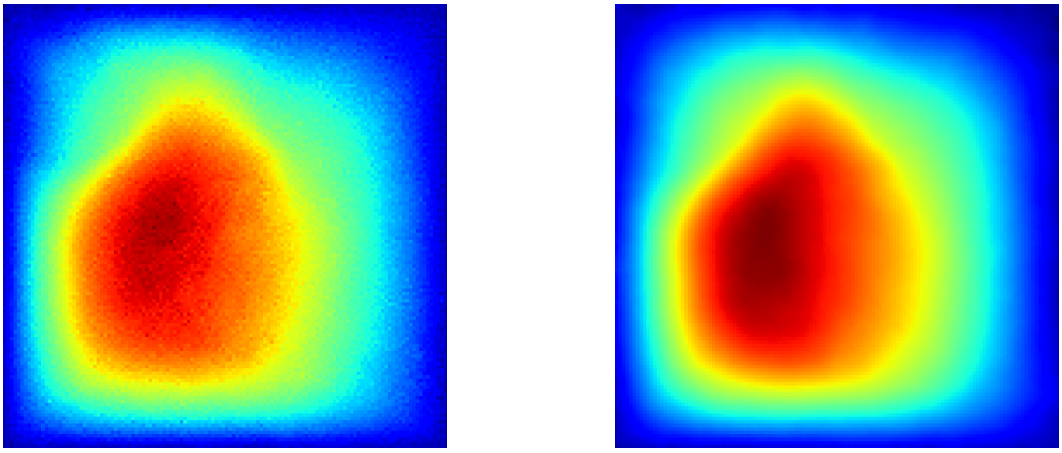

Figure 4: Qualitative comparison of unguided Darcy endpoints after Stage 2 training with L^2 (left) and L^∞ (right) residual penalties.

E.3 Matched-Duration Control Without Physics

Stage 2 differs from Stage 1 both in the residual loss and in the additional 10,000 kimg of training. To separate the two effects, we continue the Stage 1 checkpoint for the same 10,000 kimg with $\lambda_{\text{pde}} = 0$ (SplitMeanFlow only) and evaluate it with the guided protocol of Table 1. Such controls exist for Poisson and Darcy (Table 8); for Helmholtz, the remaining family in Table 2, no physics-free continuation was trained, so its Stage 2 differences cannot be separated from the effect of longer training. The residual loss accounts for a forward gain on both PDEs (small on Poisson, about 16% relative on Darcy), for the Poisson inverse gain, and for the Darcy inverse degradation, consistent with Table 2.

Table 8: Matched-duration control: target relative error in percent (mean $\pm$ standard deviation over 1000 test instances, $N = 2000$) for the Stage 1 checkpoint continued to 20,000 kimg without and with the residual loss (the latter is PIMF-FMinit in Table 1).

Task	SplitMeanFlow only	SplitMeanFlow + residual (Stage 2)
Poisson forward (u)	1.72 ± 0.84	$\mathbf{1.66 \pm 0.84}$
Poisson inverse (a)	14.78 ± 5.15	$\mathbf{13.99 \pm 4.21}$
Darcy forward (u)	4.99 ± 5.37	$\mathbf{4.18 \pm 2.06}$
Darcy inverse (a)	$\mathbf{7.86 \pm 5.33}$	9.33 ± 6.10

F RUNTIME AND COMPLEXITY

Table 9: Inference timing for Helmholtz forward reconstruction at batch size one and $N = 2000$ on one NVIDIA A100-SXM4-40GB, averaged over five instances.

Method	Seconds/instance	Peak GB	Network evaluations
DiffusionPDE	459.45 ± 17.05	3.74	4000
PIDM	447.63 ± 2.42	3.74	4000
FmPDE	189.80 ± 0.05	5.14	4000
PIMF	366.87 ± 5.16	5.14	4000
PIMF-FMinit	361.31 ± 1.44	5.14	4000

Let C_θ denote one network evaluation, $C_{\mathcal{M}}$ observation evaluation and backpropagation, and $C_{\mathcal{F}}$ residual evaluation and backpropagation. With $N_{\mathrm{pde}} \leq N$ residual-guided iterations, PIMF costs

$$N(2C_\theta + C_{\mathcal{M}}) + N_{\mathrm{pde}} C_{\mathcal{F}}. \tag{14}$$

This expression accounts separately for the endpoint and finite-interval transport evaluations. The guided baselines have the same form. DiffusionPDE and PIDM use a second-order sampler whose first denoiser evaluation also provides the Tweedie estimate, and FmPDE evaluates the velocity once for the endpoint estimate and once for the transport step, so all three cost $N(2C_\theta + C_{\mathcal{M}}) + N_{\mathrm{pde}} C_{\mathcal{F}}$ as well, with 4000 evaluations at $N = 2000$ (Table 9). PIDDM optimizes the latent instead: each of its N Adam steps evaluates the one-step student once and backpropagates the observation and residual terms, giving $N(C_\theta + C_{\mathcal{M}} + C_{\mathcal{F}})$. PIDDM is absent from Table 9 because its wall-clock time was not measured under that protocol. Network-evaluation counts do not capture the different backward graphs, residual operators, memory use, or checkpoint-loading overhead.

The wall-clock axis of Figure 7 is an estimate obtained by rescaling iterations with the per-iteration cost of Table 9. Direct end-to-end measurements at two reduced budgets (one instance, same GPU) confirm it: on Helmholtz forward recovery, PIMF reaches a 2.3% relative error at $N = 200$ in 51 s, compared with 103 s for FmPDE, which needs $N = 1000$ to reach 2.6% (at $N = 200$ FmPDE takes 34 s but remains at 14%), and 238 s for DiffusionPDE, which is still at 18% at $N = 1000$. Errors are the 100-instance means of Figure 6.

G Additional Trajectory and Sampling-Budget Figures

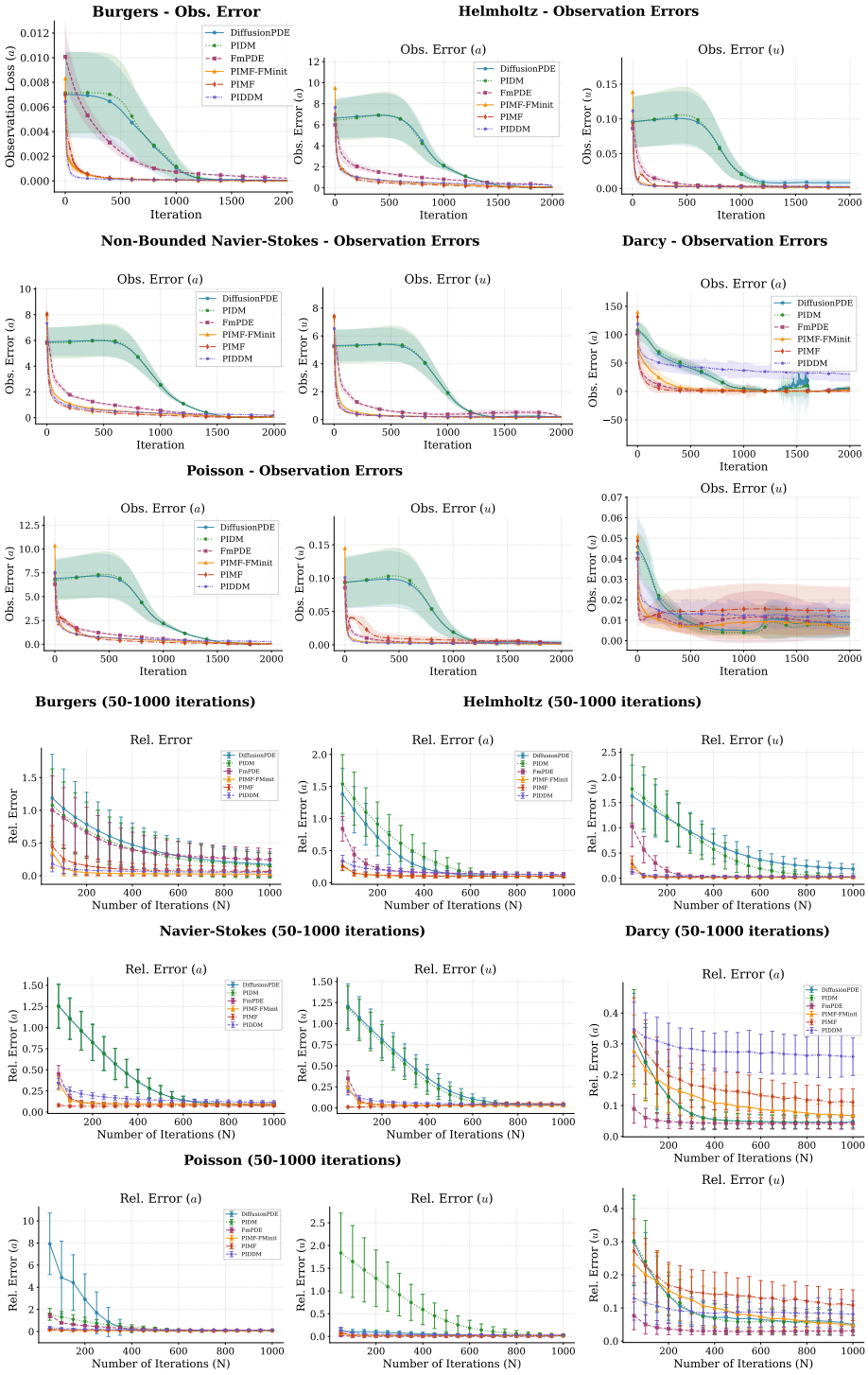

Figure 5: **Cross-family guidance and budget results** (100 instances; shading/error bars are ± 1 standard error). **Top:** observation-error trajectories at $N = 2000$. PIMF acts earlier on Helmholtz, Poisson, and NSNB, with weaker separation on Darcy and Burgers. **Bottom:** relative error for $N = 50, 100, \ldots, 1000$. Reduced-budget gains are strongest on Helmholtz and Poisson and weaker on Darcy.

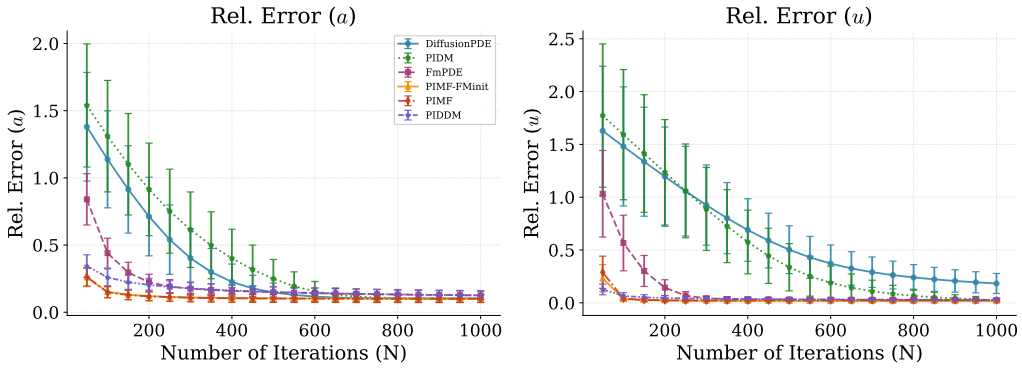

Figure 6: **Helmholtz forward reconstruction at reduced iteration budgets** (100 instances; error bars are ± 1 standard error). The coefficient a is sparsely observed and the solution u is the target field. PIMF remains accurate at smaller budgets.

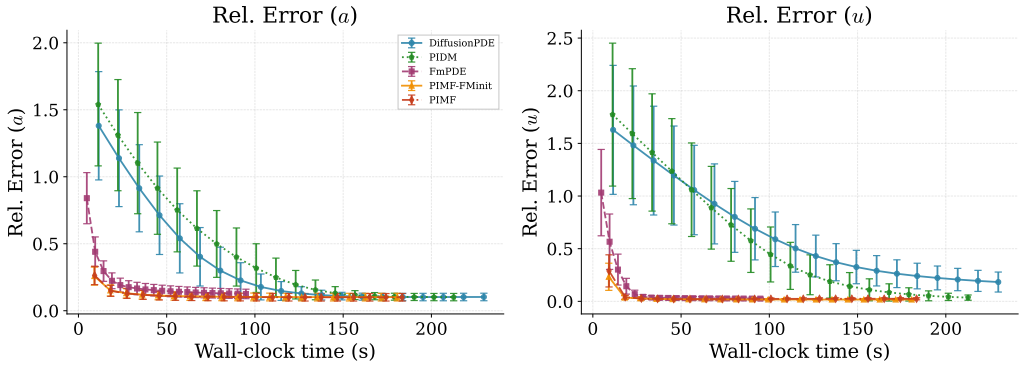

Figure 7: **Helmholtz forward reconstruction against wall-clock time.** The curves of Figure 6 with the iteration axis rescaled by each method's measured seconds per iteration (Table 9; an estimate, see the direct measurements above). FmPDE has the cheapest iteration, and PIMF the lowest error at a given wall-clock time in the low-budget regime.

H Additional Multi-Start Analysis

H.1 Novel-Solution Diagnostics

Here, novel denotes candidates that differ under the same observation condition; it does not mean that solution discovery is unique to PIMF or that a candidate is absent from the training distribution. Every PIMF result in this appendix (Figures 8 and 9 and Tables 10 to 13) uses the PIMF-FMinit checkpoint of Table 1. To distinguish structural alternatives from small perturbations, we compare candidates in the joint coefficient–solution space using correlation distance,

$$d(x, x') = 1 - \mathrm{corr}(x, x'), \tag{15}$$

and apply agglomerative hierarchical clustering with average linkage. Each cluster representative summarizes a distinct structural candidate under this descriptive criterion. The resulting groups are not interpreted as calibrated posterior modes.

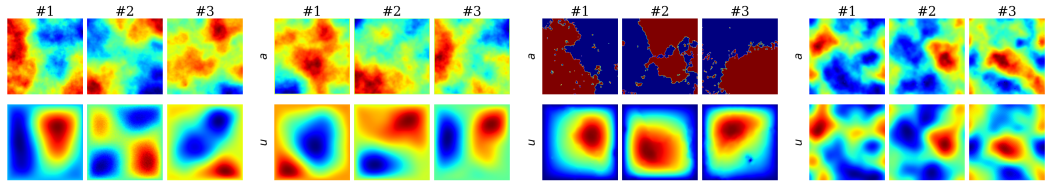

Figure 8: Cluster representatives obtained from the $M = 10$ ensemble members. Each column shows a structurally distinct solution for one representative instance per PDE.

We separately check whether the candidates retain the observations and have PDE residuals comparable to unguided MeanFlow endpoints. Table 10 reports these quantities alongside reduced- and full-budget unguided references.

Table 10: Observation and PDE consistency of multi-start candidates ($M = 10$, $N = 200$; median over seeds and instances) compared with unguided MeanFlow endpoints.

PDE	Candidate residual	Obs. error	Unguided ($N = 200$)	Unguided ($N = 2000$)
Poisson	1.3×10^{-2}	0.038	2.7×10^{-2}	5.0×10^{-3}
Helmholtz	1.0×10^{-2}	0.043	2.6×10^{-2}	5.9×10^{-3}
Darcy	1.5×10^{-1}	< 0.001	1.1×10^{-1}	1.1×10^{-2}
NSNB	1.5×10^{-4}	0.186	1.6×10^{-4}	1.6×10^{-4}

The finite-difference residual only checks each point locally. A stricter test is to ask whether a candidate's solution actually solves its own coefficient. For every candidate a_c we solve the discretized forward problem $u^\star = \mathcal{S}(a_c)$ with the same stencil as the residual loss (for Poisson and Helmholtz this is exactly how the data were generated) and measure $\|u_c - u^\star\|_2 / \|u^\star\|_2$ (Table 11).

On Poisson and Helmholtz, the candidates are far from the ground truth but usually close to their own forward solve, with a few larger outliers. So the ensemble is producing different, nearly self-consistent (a, u) pairs, not noisy copies of the ground truth. On Darcy the picture is mixed. Even the ground-truth pairs have a small nonzero distance here, because the benchmark's Darcy solver (`solve_gwf` in the DiffusionPDE data-generation code) works on a spline-interpolated nodal grid, whereas we apply the stencil directly to the stored fields. Darcy candidates sit well above that floor, and many of them are not consistent solutions at all. Table 10 does not show this, since the candidate residuals there match the unguided references on all three PDEs. NSNB is left out because its proxy residual has no forward solver.

Table 11: Forward-solve consistency of the multi-start candidates: relative distance between each candidate solution u_c and the solution $\mathcal{S}(a_c)$ of its own coefficient (median and maximum over 50 candidates: five instances, $M = 10$ each), the same quantity for the ground-truth pairs (floor of the check), and the median relative distance of u_c to the ground-truth u. On Darcy, 42% of candidates exceed 0.1.

PDE	Median $\|u_c - \mathcal{S}(a_c)\|/\|\mathcal{S}(a_c)\|$	Max	Ground-truth floor	Median $\|u_c - u\|/\|u\|$
Poisson	0.015	0.16	$< 10^{-13}$	0.90
Helmholtz	0.014	0.07	$< 10^{-13}$	0.67
Darcy	0.055	0.94	0.015	0.34

H.2 Ensemble Dispersion Metrics

Figure 9 complements the quantitative diagnostics below by showing where ensemble members disagree spatially for one representative instance per PDE.

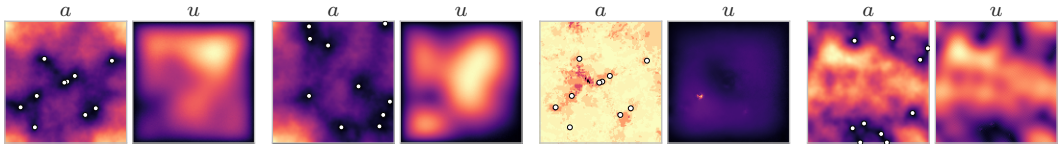

Figure 9: **Cross-family empirical ensemble variance.** Coefficient (left) and solution (right) variance maps across $M = 10$ PIMF-FMinit reconstructions with $N = 200$ sampling steps for one representative forward instance per PDE. White markers indicate observed coefficient locations.

On five forward instances per family, each with ten coefficient observations, we compare $M = 50$, $N = 200$ PIMF and DiffusionPDE ensembles with a 50-member ensemble of physics-informed neural networks (PINNs) and preconditioned Crank–Nicolson (pCN) MCMC, all on the same instances and observation locations. For the learned methods, we match ensemble size rather than aggregate compute. We report CRPS divided by the MAE of the ensemble mean (below one means the ensemble improves over a point mass at its mean), 80% marginal interval coverage (nominal 0.804 for the finite-ensemble convention), and spread–skill ratio (near one means matched spread and error scale).

PINN ensemble. For each instance, we train 50 members from different random initializations. Each member uses two Fourier-feature MLPs (four hidden layers of width 256) that map (x, y) to a and u, trained with Adam for 20,000 iterations on the ten coefficient observations, a zero boundary condition on u, and the finite-difference PDE residual of Table 4.

pCN. pCN samples a under the Gaussian random field prior that generated the datasets, so it uses the exact prior that PIMF and DiffusionPDE learn from data. The likelihood uses only the ten observed coefficient values with a small Gaussian tolerance, and u is obtained by a forward solve for each sample. Per instance, we run two chains of 200,000 steps (half burn-in, step size adapted to 20–30% acceptance) and keep 25 evenly spaced states per chain. Because the likelihood depends only on a ten-dimensional projection of the latent, each retained sample keeps the chain's component in that subspace and draws the rest exactly from the prior. Split-$\hat{R}$ is at most 1.011 on all instances.

Table 12: Dispersion metrics (coefficient/solution, a/u).

PDE	Method	CRPS/MAE	80% coverage	Spread–skill
Poisson	PIMF	$0.709/0.748$	$0.877/0.934$	$1.15/1.50$
	DiffusionPDE	$0.734/0.764$	$0.932/0.963$	$1.38/1.61$
	PINN ensemble	$0.748/0.923$	$0.580/0.129$	$0.66/0.12$
	pCN	$0.699/0.688$	$0.830/0.893$	$1.01/1.14$
Helmholtz	PIMF	$0.712/0.689$	$0.797/0.792$	$0.96/0.95$
	DiffusionPDE	$0.726/0.725$	$0.770/0.720$	$1.03/0.83$
	PINN ensemble	$0.767/0.941$	$0.529/0.131$	$0.60/0.09$
	pCN	$0.706/0.709$	$0.809/0.834$	$0.94/1.20$
Darcy	PIMF	$0.447/0.723$	$1.000/0.947$	$1.11/1.62$
	DiffusionPDE	$0.469/0.630$	$1.000/0.732$	$1.07/1.00$
	PINN ensemble	$0.940/0.900$	$0.004/0.148$	$0.12/0.15$
	pCN	$0.523/0.699$	$0.976/0.708$	$0.91/0.81$

PIMF and DiffusionPDE are broadly comparable. PIMF is close to nominal coverage on Helmholtz; Poisson and Darcy show overcoverage or excess spread. The PINN ensemble is strongly under-dispersed, whereas pCN varies by task. These descriptive diagnostics do not establish posterior calibration.

H.3 Spatial Disagreement and Reconstruction Error

For the ten-member runs, we also compare per-pixel ensemble standard deviation $\sigma(c)$ with the absolute ensemble-mean error $|\bar{x}_0(c) - x_{\mathrm{GT}}(c)|$. Table 13 pools five instances per family and reports the association without pixel-level significance tests, because spatial dependence prevents treating all pooled pixels as independent replicates.

Table 13: Descriptive spatial association between ensemble disagreement and absolute reconstruction error. Pearson r and Spearman ρ are pooled over five instances per PDE; u is unobserved.

	Channel a		Channel u	
PDE	r	ρ	r	ρ
Poisson	0.433	0.409	0.540	0.576
Helmholtz	0.478	0.474	0.618	0.637
Darcy	0.518	0.660	0.372	0.440

Positive correlations indicate that disagreement concentrates where the ensemble mean is less accurate. With five instances per family and pooled spatial locations, this remains a descriptive error-localization diagnostic.

I PIDDM BASELINE: DISTILLATION AND TUNING PROTOCOL

PIDDM (Zhang & Zou, 2025) distils a pretrained generative teacher into a one-step student $d_\theta(\epsilon) \mapsto x_0$ and solves inverse problems by optimizing the latent ϵ at test time. The authors have not released code, and the paper reports results on a different set of PDEs, so we re-implemented the method on our benchmarks and followed the paper as closely as we could. This section records what we did, so that the comparison in Table 1 can be judged and reproduced.

Teacher and pair bank. The teacher is the pretrained Flow Matching checkpoint that also initializes PIMF-FMinit, so both methods start from the same model. For each PDE we generate a fixed bank of $32{,}768$ training pairs $\{\epsilon, x_0\}$ and 512 held-out pairs by integrating the teacher's probability-flow ODE for 100 Euler steps from each ϵ. Distillation is then plain supervised regression on this fixed dataset. Every choice of λ and every seed sees the same targets, so differences between runs come from the loss and not from teacher sampling noise.

Distillation objective. The student uses the same SongUNet backbone as the teacher but is trained from scratch: the teacher predicts a velocity and the student predicts a sample, so the teacher weights are not a useful initialization. It minimizes

$$\mathcal{L}(\theta) = \big\|d_\theta(\epsilon) - x_0\big\|_2^2 + \lambda_{\text{train}}\,\mathcal{R}\big(d_\theta(\epsilon)\big), \tag{16}$$

where $\mathcal{R}$ is the L^2 PDE residual of the decoded sample in physical units. We train for 2 Mimg with Adam, learning rate 10^{-4}, and batch size 32 on 4 GPUs, and ramp λ_{train} linearly from 0 over the first 50 kimg so that the student first learns the regression map.

Choosing λ_{train}. Residual scales differ by orders of magnitude across PDEs, so a single shared λ would not mean the same thing on each of them. For each PDE we first train a $\lambda = 0$ student, read off its held-out sample loss and held-out residual at the best tick, and set

$$\lambda_{\text{train}} = \rho \cdot \frac{\text{held-out sample loss}}{\text{held-out PDE loss}}, \qquad \rho = 1. \tag{17}$$

This is the weight at which the physics term contributes as much as the regression term for the $\lambda = 0$ gate student. We measure the ratio on the held-out pairs rather than on the training bank, because a student that memorizes the bank drives its training loss far below its real distillation error.

Snapshot selection. The original method early-stops on the observation loss. We instead train for the full budget and keep the tick with the lowest held-out sample loss on the EMA weights. This is the same criterion without a patience hyperparameter, and the held-out set stops the selection from rewarding memorization.

Test-time optimization. We follow Algorithm 3 of Zhang & Zou (2025): starting from $\epsilon \sim \mathcal{N}(0, I)$, we run $N = 2000$ Adam steps on

$$\frac{1}{|\mathcal{M}|}\big\|\big(d_\theta(\epsilon) - x_{\text{obs}}\big) \odot \mathcal{M}\big\|_2^2 + \lambda_{\text{infer}}\,\mathcal{R}\big(d_\theta(\epsilon)\big), \tag{18}$$

and report $x_{\text{mix}} = x_{\text{obs}} \odot \mathcal{M} + d_\theta(\epsilon) \odot (1 - \mathcal{M})$. The 2000 steps match the sampling budget of the other generative baselines in Table 1, and the observations are the same (500 scattered points; 5 observed columns for Burgers).

We evaluate the residual on the raw decoded sample x rather than on the spliced x_{mix}. With scattered sensors, the hard-replaced pixels look like step discontinuities to a finite-difference stencil, so the residual of x_{mix} measures splice artifacts rather than physics. On Poisson inverse we measure $\mathcal{R}(x_{\text{mix}}) \approx 4 \times 10^4$ against $\mathcal{R}(x) \approx 1.5 \times 10^{-3}$, and the resulting gradient drowns the physics signal. The reconstruction errors in Table 1 are computed on x_{mix}, the returned reconstruction.

Tuning. For each PDE and direction we sweep $\lambda_{\text{infer}} \in \{0, 0.1, 1, 10, 100, 1000\} \times \lambda_{\text{train}}$ and the Adam learning rate $\in \{3 \times 10^{-3}, 10^{-2}\}$ on ten tuning instances at offsets 1000–1009, disjoint from the 1000 test instances at offsets 0–999. NSNB only has 1000 instances in total, so its tuning offsets are 0–9 and overlap with the test set; this overlap can only favor the baseline. Table 14 lists the selected settings. The inference-time residual helps most in the forward direction on Poisson and Helmholtz, where the largest λ_{infer} wins; a moderate weight (10–100 $\times\ \lambda_{\text{train}}$) is selected for Darcy forward and for Poisson and Helmholtz inverse, and the residual is neutral or harmful on Darcy inverse, NSNB, and Burgers, where a small or zero weight is selected.

Table 14: PIDDM settings. λ_{train} is derived from the $\lambda = 0$ gate run at $\rho = 1$; the inference learning rate and λ_{infer} are selected on the tuning offsets. Burgers is a single-field task with one direction.

		Forward		Inverse	
PDE	λ_{train}	lr	λ_{infer}	lr	λ_{infer}
Darcy	1.15×10^{-4}	10^{-2}	1.15×10^{-2}	3×10^{-3}	1.15×10^{-5}
Poisson	1.09×10^{-4}	3×10^{-3}	1.09×10^{-1}	10^{-2}	1.09×10^{-3}
Helmholtz	1.41×10^{-4}	3×10^{-3}	1.41×10^{-1}	10^{-2}	1.41×10^{-2}
NSNB	4.38×10^{-2}	3×10^{-3}	4.38×10^{-2}	10^{-2}	0
Burgers	2.36	10^{-2}	0	–	–

J PyTorch-Style Pseudocode

The sketch below mirrors Algorithms 1 and 2 on a batch. Each sample draws its own interval (r, t), whereas the training branch (flow matching or interval splitting) is drawn once per batch and shared by all samples. The Stage 2 residual is evaluated only on the gated samples, so the endpoint hop costs one gradient-free network evaluation per gated sample; its L^∞ norm is taken jointly over all gated samples and grid points (the batch maximum), so gated samples do interact through this term. At inference, the observation and PDE terms are summed into one loss before differentiation; because the gradient is linear, this gives the same update as Algorithm 1. Inference norms are taken per sample and summed, so samples in a batch do not interact there.

```
import torch

# m_theta(x, r, t): MeanFlow network, mean velocity of x over [r, t]
# F(x):             PDE residual of x = (a, u) on the 128 x 128 grid
# D(x):             channel-wise map from model space to physical units
# x0 = (a, u):      clean coefficient-solution pair, x1: Gaussian noise
# q_fm = 0.6, lambda_pde = 1e-3, r_max = 0.15, S: total updates

def l2(z):    # per-sample L2 norm, summed over the batch
    return z.flatten(1).norm(dim=1).sum()

# ------------------------------------------------------------
# Training: two-stage Physics-Informed MeanFlow
# ------------------------------------------------------------

for k, x0 in enumerate(dataloader):
    B   = x0.shape[0]
    x1  = torch.randn_like(x0)
    t   = torch.rand(B, 1, 1, 1)
    x_t = (1 - t) * x0 + t * x1                # linear path

    # One branch per batch (drawn on rank 0 and broadcast):
    # interval splitting (r < t) or flow matching (r = t).
    is_split = bool(torch.rand(()) >= q_fm)
    r = t * torch.rand(B, 1, 1, 1) if is_split else t

    m_pred = m_theta(x_t, r, t)                # carries the gradient

    if is_split:
        # SplitMeanFlow target: two gradient-free evaluations, no JVP.
        with torch.no_grad():
            gamma = torch.rand(B, 1, 1, 1)
            s     = (1 - gamma) * t + gamma * r
            u2    = m_theta(x_t, s, t)
            x_s   = x_t - (t - s) * u2         # rollback t -> s
            u1    = m_theta(x_s, r, s)
            m_tgt = (1 - gamma) * u1 + gamma * u2
    else:
        m_tgt = x1 - x0
```

```
    # Per-sample mean squared error over all entries, shape (B,).
    loss = (m_pred - m_tgt).pow(2).flatten(1).mean(dim=1)

    # Stage 2: PDE residual on the endpoint reached from x_r, gated
    # to samples of a splitting batch that start close to the data.
    g = (r < r_max).flatten()
    if k >= S // 2 and is_split and g.any():
        # First hop reuses m_pred and carries the gradient.
        x_r = x_t[g] - (t[g] - r[g]) * m_pred[g]
        # Endpoint hop over (0, r) under a stop-gradient.
        with torch.no_grad():
            m_end = m_theta(x_r, torch.zeros_like(r[g]), r[g])
        x_hat_0 = x_r - r[g] * m_end
        # One L-infinity norm over all gated samples and grid points,
        # added to the loss of each gated sample.
        res  = F(x_hat_0).abs().max() / 128**2
        loss = loss + lambda_pde * res * g.float()

    loss = loss.mean()
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
    ema.update(m_theta)

# ------------------------------------------------------------
# Inference: guided PIMF sampling
# ------------------------------------------------------------

# obs[c] = (mask, y): sensor mask, zero-filled data of channel c
#                     in physical units
# zeta_obs[c], zeta_pde: guidance weights
# delta_pde, alpha_obs: guidance schedule

def guided_pimf(m_theta, obs, zeta_obs, zeta_pde, shape, N=2000,
                delta_pde=0.8, alpha_obs=0.1):
    t_grid = torch.linspace(1.0, 0.0, N + 1)  # t_0 = 1, t_N = 0
    i_pde  = int(delta_pde * N)
    x      = torch.randn(shape)

    for i in range(N):
        t_i, t_next = t_grid[i], t_grid[i + 1]
        x = x.detach().requires_grad_(True)

        # MeanFlow projection: one-step estimate of the clean state,
        # mapped to physical units.
        x_hat_0 = D(x - t_i * m_theta(x, 0.0, t_i))

        # Unsquared losses on the projection. Once PDE guidance starts,
        # the observation term is rescaled by alpha_obs.
        L_obs = sum(zeta_obs[c] * l2(mask * (x_hat_0[:, c] - y))
                    for c, (mask, y) in obs.items())
        if i <= i_pde:
            L = L_obs
        else:
            L = alpha_obs * L_obs + zeta_pde * l2(F(x_hat_0)) / 128**2
        grad = torch.autograd.grad(L, x)[0]

        # Transport step with the finite-interval mean velocity.
        with torch.no_grad():
            x_bar = x - (t_i - t_next) * m_theta(x, t_next, t_i)
        x = x_bar - grad

    return x.detach()
```

K QUALITATIVE RECONSTRUCTIONS

Each gallery shows five fixed test instances, the sparse observation, and reconstructions from the methods available in the original visualization pipeline. Every image is included at its native aspect ratio.

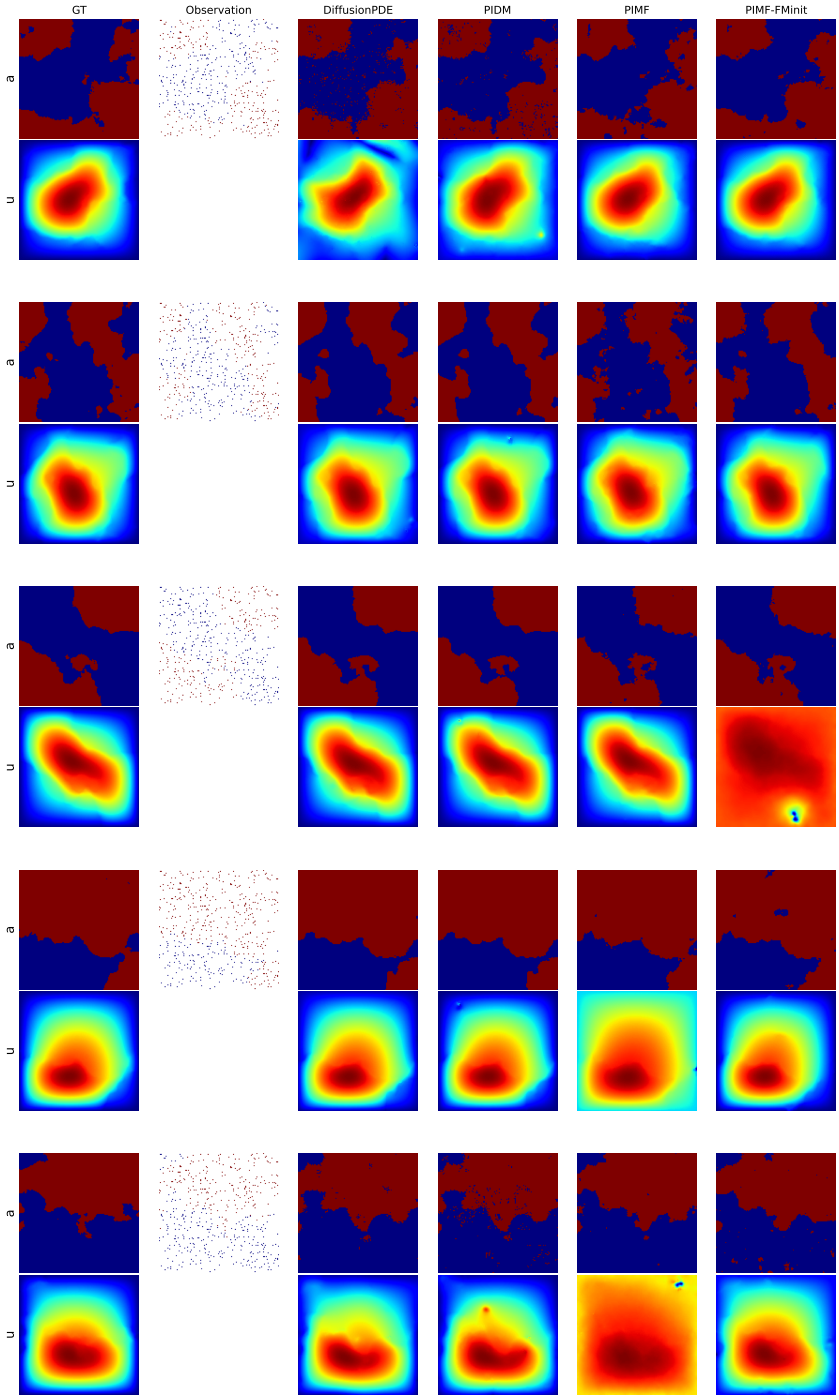

Figure 10: Five examples for Darcy forward recovery.

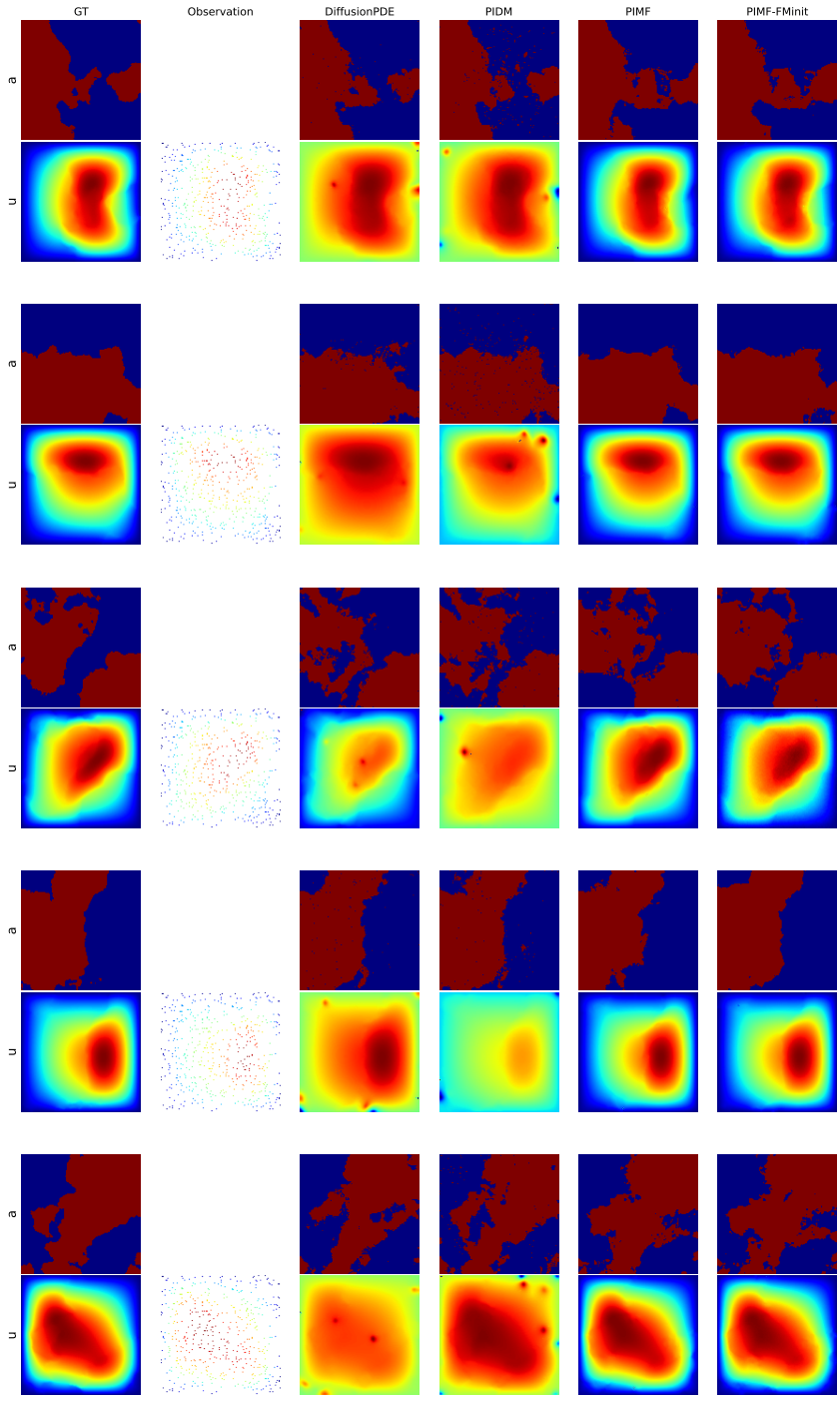

Figure 11: Five examples for Darcy inverse recovery.

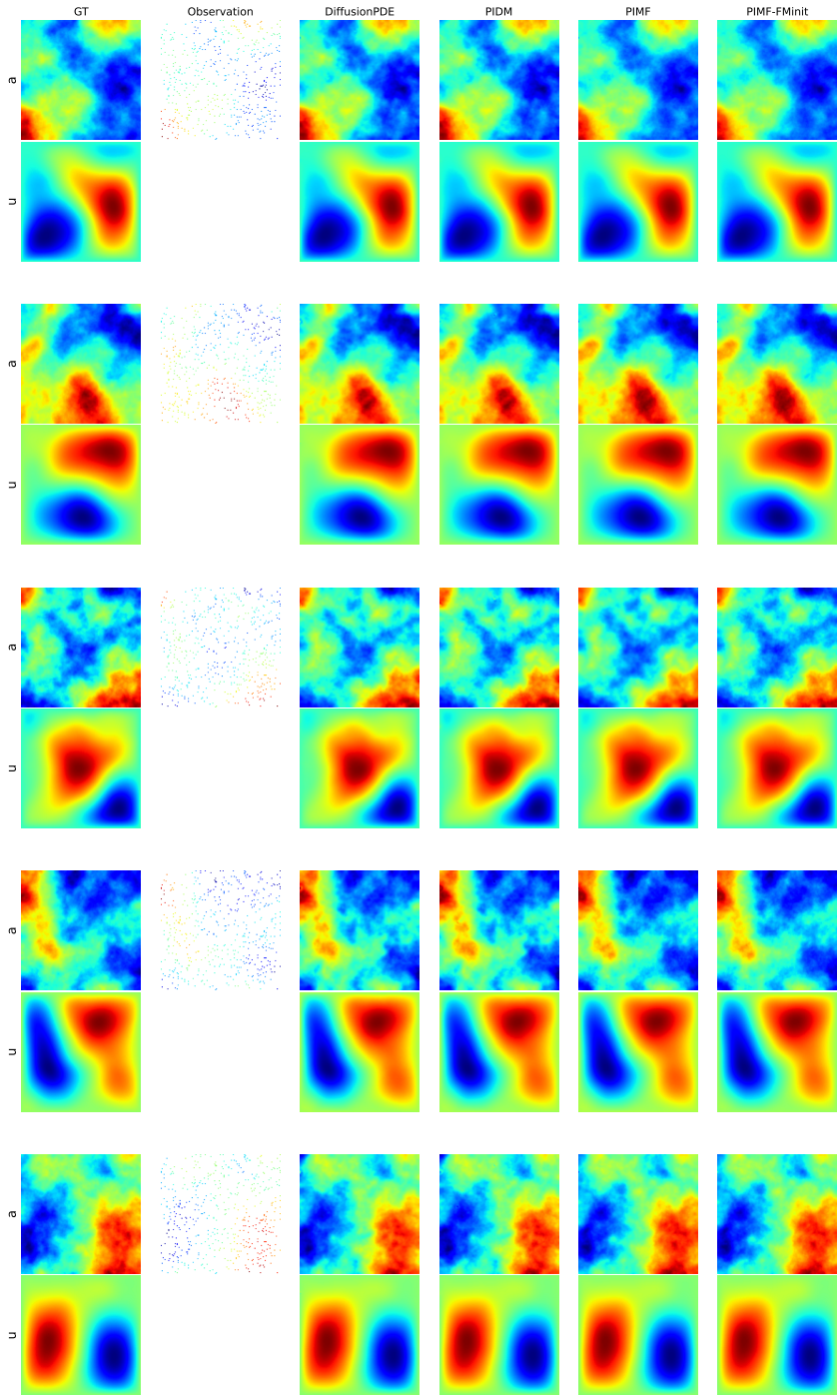

Figure 12: Five examples for Poisson forward recovery.

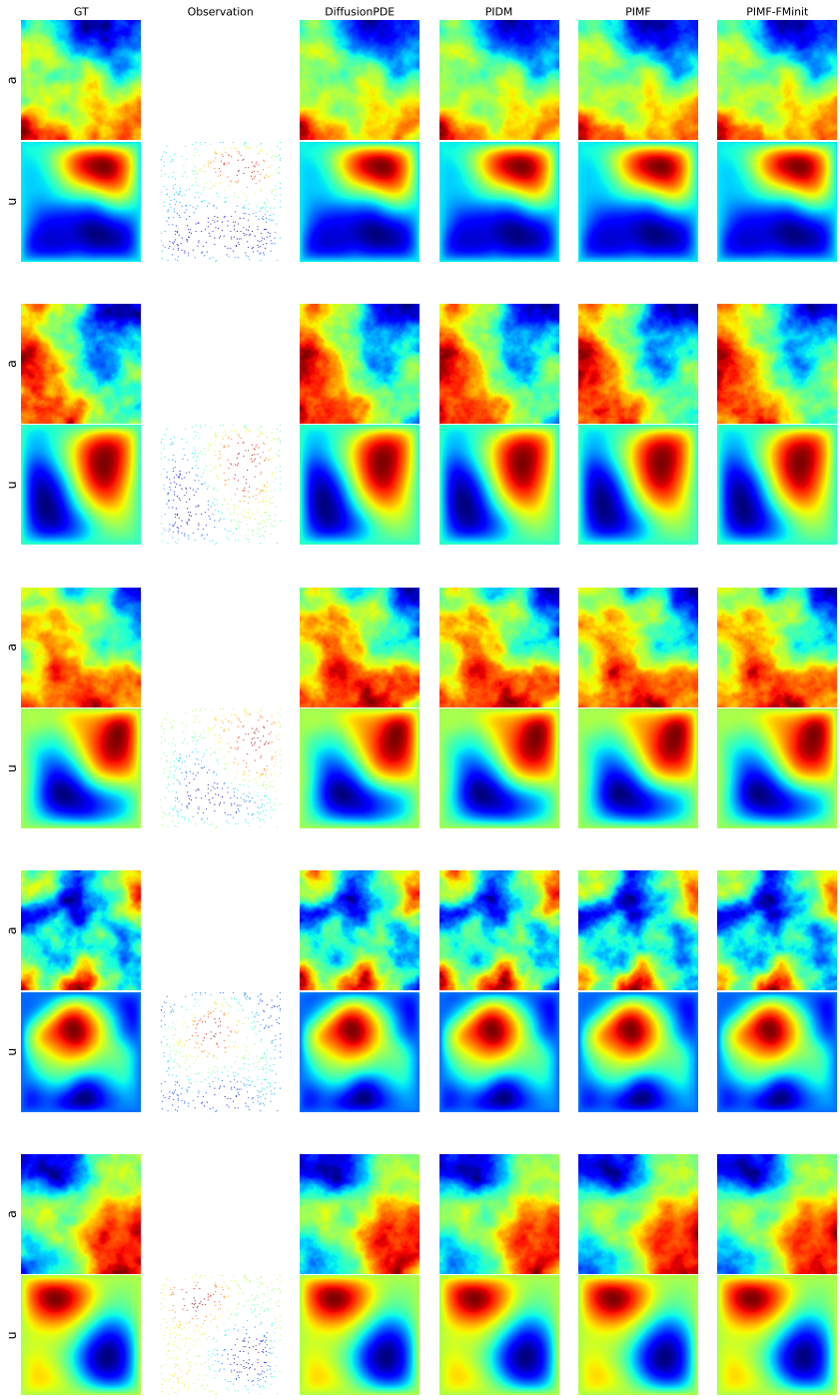

Figure 13: Five examples for Poisson inverse recovery.

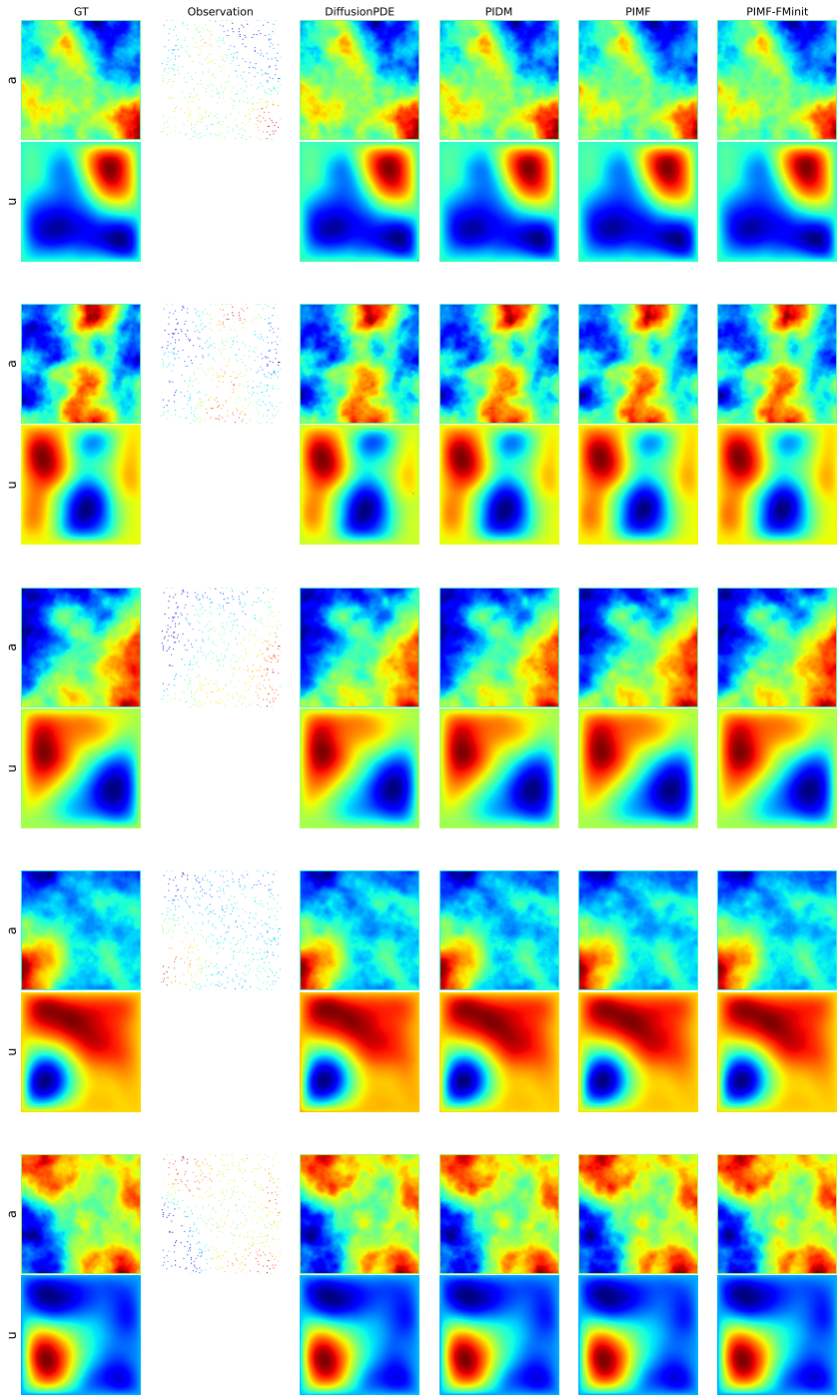

Figure 14: Five examples for Helmholtz forward recovery.

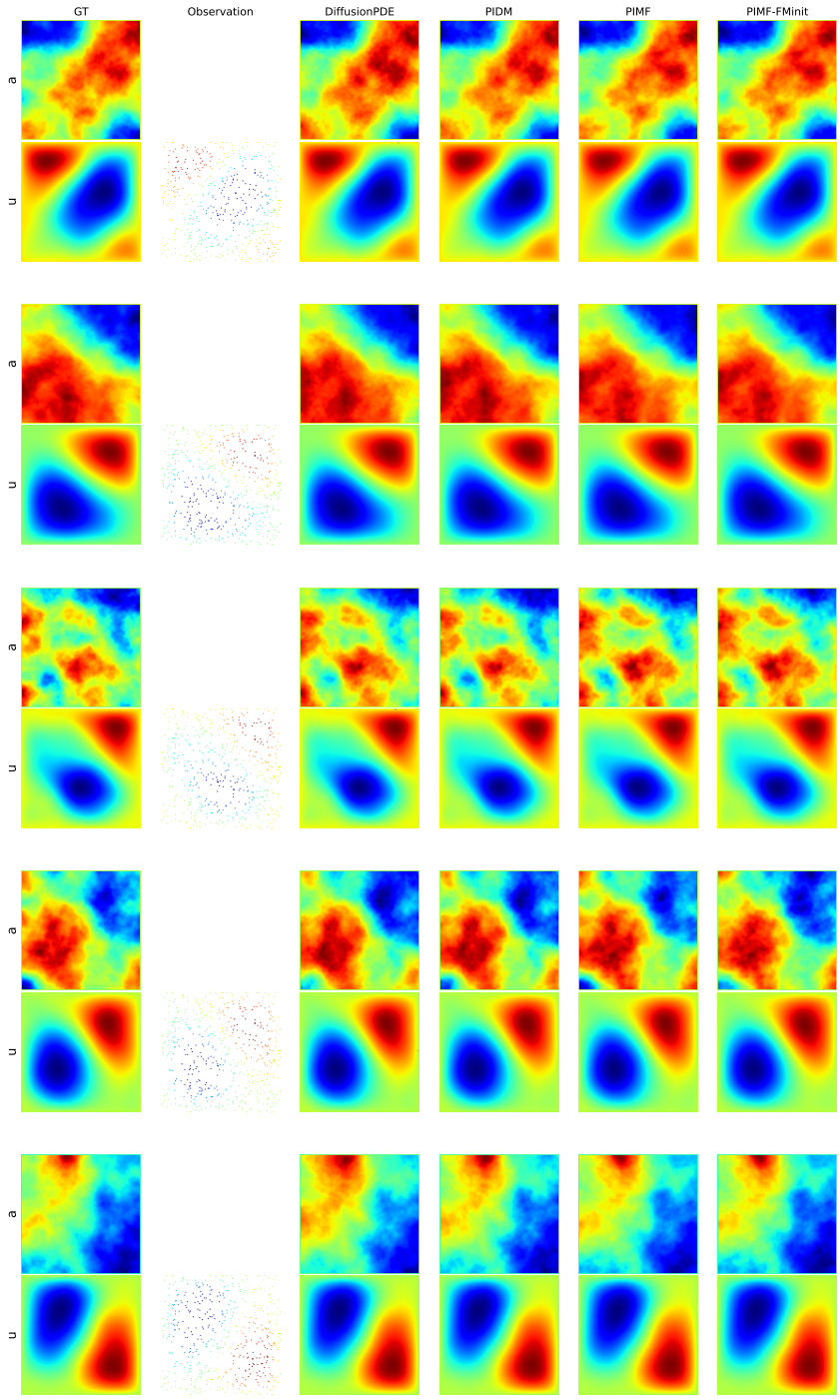

Figure 15: Five examples for Helmholtz inverse recovery.

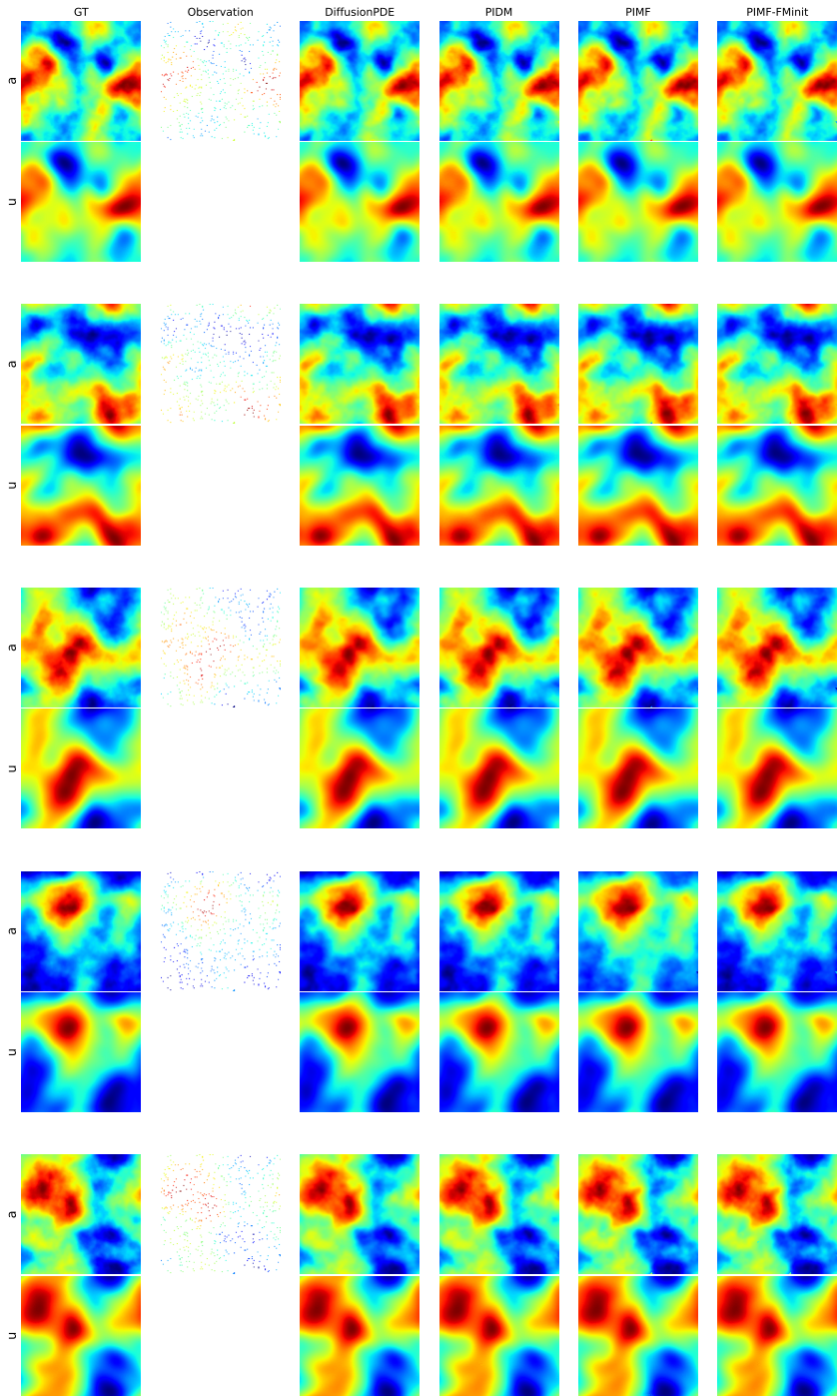

Figure 16: Five examples for NSNB forward recovery. These images support reconstruction comparisons only.

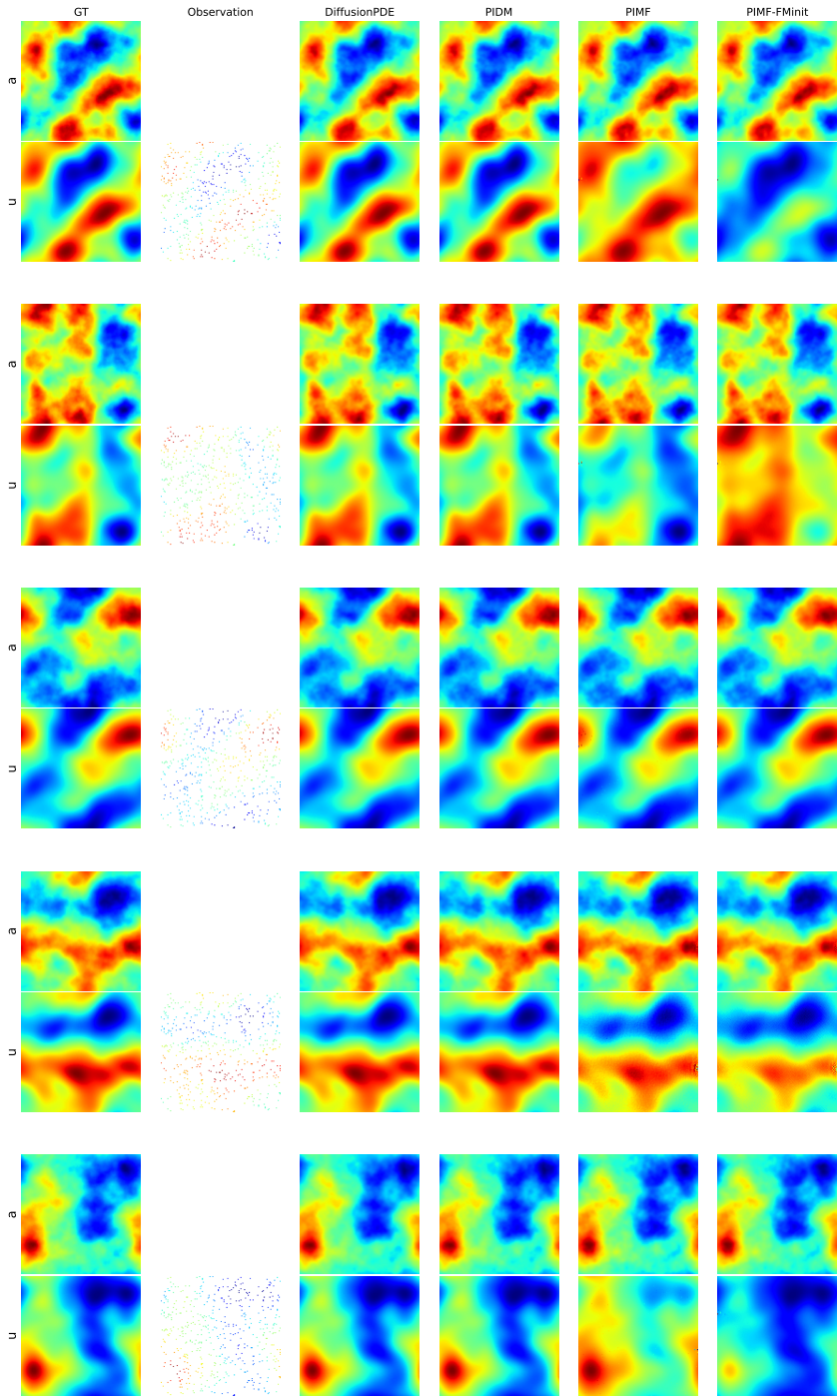

Figure 17: Five examples for NSNB inverse recovery. These images support reconstruction comparisons only.

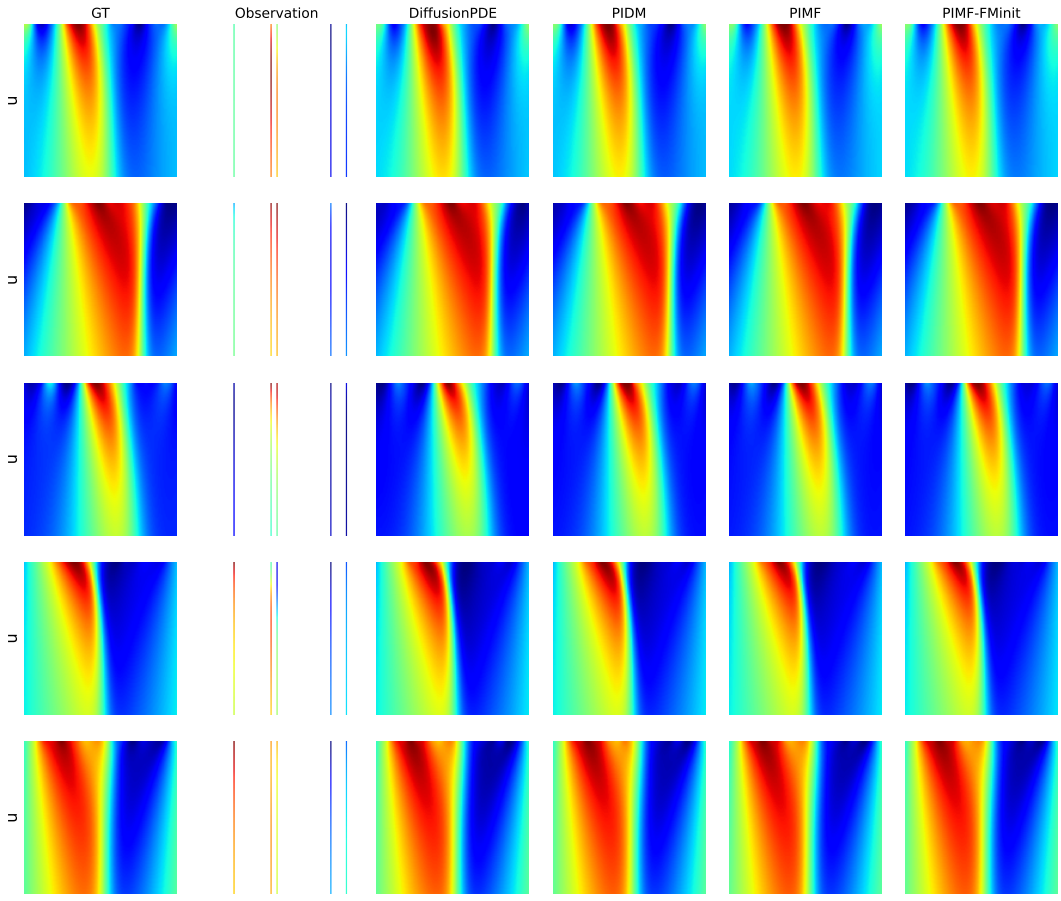

Figure 18: Burgers space–time trajectory reconstruction on five test instances.