OCL-PDE: A GENERATIVE FRAMEWORK FOR PDE INVERSE PROBLEMS WITH OBSERVATION-COMPLEMENTARY LATENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Partial differential equation (PDE) inverse problems are often ill-posed, making fine-scale details difficult to recover. We address this problem by introducing a learned observation-complementary latent representation that preserves reconstruction-relevant information and is combined with the observation to reconstruct the unknown field. Building on this representation, we propose OCL-PDE, a generative framework that encourages the observation to guide large-scale structure and the latent to supply complementary fine-scale details. OCL-PDE is built on a physics-aware autoencoder (AE) and conditional Flow Matching, supporting inverse reconstruction as well as forward PDE prediction. Experiments demonstrate improved reconstruction accuracy and fine-detail recovery compared with the evaluated baselines.

1 INTRODUCTION

Inverse problems governed by partial differential equations (PDEs) seek to recover unknown coefficients, initial or boundary conditions, or source terms from indirect observations (Isakov, 2017). Given an unknown field c and observations y produced through a PDE forward process, the task is to infer c from y. The available observations may be coarse, sparse, or noisy; in time-dependent problems, dissipative evolution may further erase information about the initial state. Such problems are often ill-posed: distinct fields can produce similar observations, while local variations or interfaces may be weakly reflected in the measured response (Isakov, 2017; Stuart, 2010). Consequently, an inverse method may agree with the observations while failing to recover important structures in the underlying field.

These difficulties highlight the need for prior information about the unknown field to guide reconstruction. Generative models offer a data-driven way to learn such information by modeling distributions of physical fields. Existing approaches exploit this idea through joint modeling of coefficient and solution fields or through learned latent representations (Huang et al., 2024; Zang & Koutsourelakis, 2025; Wang et al., 2026). However, learning a low-dimensional representation alone does not ensure accurate reconstruction in latent-space inverse methods. The representation must retain information useful for recovering field details, and useful latent variables must be inferred from the available observations. These requirements motivate learning latent representations specifically for observation-conditioned reconstruction.

To address these requirements, we introduce a learned observation-complementary latent representation that preserves reconstruction-relevant information and is combined with observations to recover the unknown field. Observations provide a direct basis for reconstruction, while latent features contribute corrections to field structures that are not adequately recovered from observations alone. This design encourages observations to guide the large-scale structure and the latent to supply complementary information for recovering fine-scale variations.

We implement this idea by proposing a framework, OCL-PDE, which learns and generates *observation-complementary latent* (OCL) representations for inverse reconstruction. A physics-aware autoencoder learns a forward response predictor and a decoder that combines observations with a compact latent. The decoder includes an observation-based reconstruction pathway and a latent correction pathway. After the autoencoder is trained, conditional Flow Matching learns to generate

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.

its latent codes from Gaussian noise conditioned on observations. At inference, the decoder combines the generated latent with the observation to reconstruct the unknown field. The main contributions of our work are summarized as follows:

- We develop a generative framework that combines an autoencoder with Flow Matching to improve fine-scale reconstruction through a direct observation pathway and a compact latent correction pathway, while also enabling forward PDE prediction.
- We develop a two-stage learning procedure that trains the autoencoder with reconstruction supervision and PDE constraints on predicted responses, then freezes it to learn observation-conditioned latent generation through Flow Matching.
- We demonstrate strong reconstruction results on both backward Allen–Cahn recovery and Darcy coefficient inversion under coarse or sparse observations, with particularly substantial improvements over all evaluated baselines in backward Allen–Cahn recovery.
- We conduct controlled ablations which further confirm that the learned latent captures observation-complementary information beyond the observation-only pathway.

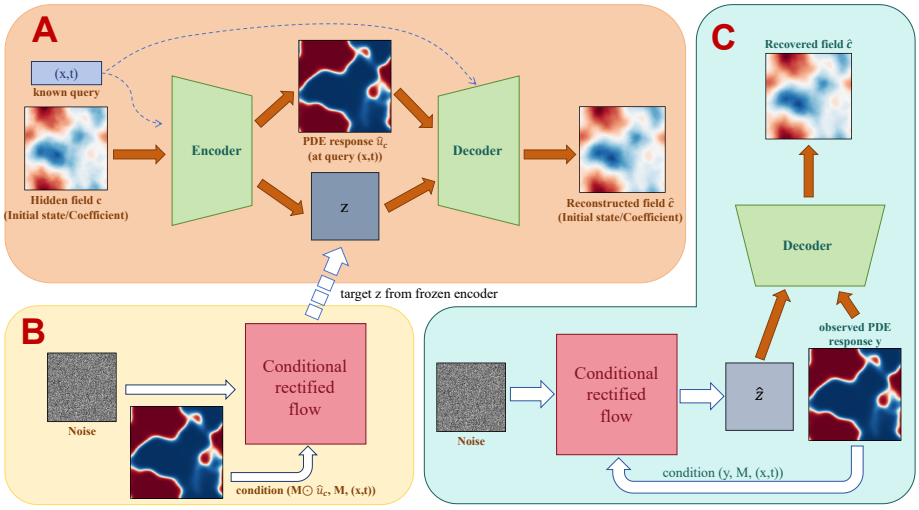


Figure 1: Overview of OCL-PDE. The framework learns observation-complementary latents and generates them conditioned on observations to recover unknown fields. (A) The physics-aware autoencoder predicts the PDE response $\widehat{u}_c$ and encodes observation-complementary information in a latent z; the decoder combines $\widehat{u}_c$ and latent z to reconstruct the input field. (B) With the autoencoder frozen, the conditional Flow learns to transform Gaussian noise into the encoder's latents z guided by the observed response. (C) At inference, the Flow generates a latent from noise conditioned on the available observation y, and the decoder combines the generated latent $\widehat{z}$ and y to recover the unknown field c.

2 Related Work

PDE inverse problems. Classical inverse formulations recover unknown fields by fitting observations subject to the governing PDE, often with regularization to stabilize recovery (Stuart, 2010). Numerical methods, including finite element and boundary element methods, have been developed to solve both forward and inverse PDE problems (Harrach, 2021; Lesnic et al., 2002). Learning-based approaches also combine observations with physical models. PINNs (Raissi et al., 2019) use observation data and PDE residuals to infer unknown parameters. The Deep Ritz method (E & Yu, 2018) approximates solutions by minimizing variational energies, while weak adversarial networks (WANs) (Zang et al., 2020; Bao et al., 2020) recover unknown coefficients and solutions from measurements using weak PDE constraints. Neural operators such as DeepONet (Lu et al., 2021) and FNO (Li et al., 2021) learn mappings between function spaces. PI-DeepONet (Wang et al., 2021) and PINO (Li et al., 2024) further incorporate PDE constraints into operator learning. These methods do not explicitly model field distributions as generative priors for inverse reconstruction, potentially limiting fine-detail recovery when observations are incomplete or degraded.

Generative priors for PDE reconstruction. Generative models provide such priors, using learned field distributions to aid reconstruction from incomplete or degraded observations. Diffusion models, including DDPM and DDIM (Ho et al., 2020; Song et al., 2021), generate samples through iterative denoising. Diffusion posterior sampling (Chung et al., 2023) incorporates measurement information into this process for inverse recovery. DiffusionPDE (Huang et al., 2024) learns a joint distribution of coefficient and solution fields, combining sparse observations and PDE constraints to guide reconstruction. Related physics-informed diffusion methods recover high-fidelity flow fields or downscale coarse PDE solutions (Shu et al., 2023; Lu & Xu, 2024), addressing spatial detail missing from low-fidelity inputs. Conditional Flow Matching has also been applied to Bayesian inverse problems to generate unknown parameters conditioned on observations (Sherki et al., 2025). These formulations generate fields or their physical parameterizations, without explicitly learning a latent representation dedicated to complementing observations for fine-detail recovery.

Learned latent representations for PDE reconstruction. An alternative is to learn compact field representations and perform inference in latent space, with VAEs providing a standard generative formulation (Kingma & Welling, 2014). FunDiff (Wang et al., 2026) combines a function autoencoder with latent diffusion to reconstruct and generate continuous fields under physical constraints. DGenNO (Zang & Koutsourelakis, 2025) jointly represents coefficient and solution fields through shared latent variables and uses physics constraints to learn from unlabeled inputs. IGNO (Bao & Zang, 2025) uses low-dimensional coefficient representations and performs inversion by optimizing latent variables against observations. These methods learn latent representations of physical fields. Our framework gives the latent z a more specific role: to preserve reconstruction-relevant information, particularly fine-scale details.

3 METHODOLOGY

Let $c \in \mathcal{C}$ denote the unknown field, such as a coefficient or initial state. y is the possibly noisy observation:

$$y = \mathcal{F}(c) + \varepsilon, \tag{1}$$

Here, $\mathcal{F} : \mathcal{C} \to \mathcal{Y}$ combines the governing PDE solution operator with the measurement process; ε is task-specific measurement noise and is zero for clean observations. The goal is to recover c from y.

Our OCL-PDE framework consists of an encoder E_{θ_E}, a decoder D_{θ_D}, and a conditional Flow Matching model v_{θ_F}. The encoder E_{θ_E} predicts the forward response $\widehat{y}$ and an observation-complementary latent z. Given an observation y, the Flow v_{θ_F} obtain the corresponding z conditioned on y. The decoder D_{θ_D} combines y with z to reconstruct c. This process is illustrated in Figure 1.

3.1 PHYSICS-AWARE AUTOENCODER AND OBSERVATION-COMPLEMENTARY LATENT

In OCL-PDE, the physics-aware autoencoder predicts the forward response and encodes the input field into an observation-complementary latent, both of which its decoder uses to reconstruct the unknown field, as shown in Figure 1A.

The encoder has a shared stem, an response branch that predicts $\widehat{y}$, and a latent branch that encodes the complete unknown field c:

$$(\widehat{y}, z) = E_{\theta_E}(c). \tag{2}$$

Forward prediction is also available from the encoder response $\widehat{y}$ when c is given.

We use a latent vector $z \in \mathbb{R}^d$, with fewer components than the discretized field c, to compress information that complements y for reconstruction. The encoder predicts a mean vector $\mu \in \mathbb{R}^d$ and a variance vector $\sigma \in \mathbb{R}^d$ to construct the latent z. Each component z_i has mean μ_i and standard deviation σ_i. z is constructed as

$$z_i = \mu_i + \sigma_i \xi_i, \qquad \xi_i \overset{\text{iid}}{\sim} \mathcal{N}(0,1), \quad i = 1, \ldots, d. \tag{3}$$

The decoder forms a base prediction and a spatial gate from the response y, its validity mask, and coordinates. A separate path injects z at multiple scales and combines it with response features to produce a residual. The two paths give

$$\widehat{c} = D_{\theta_D}(z, y) = P_{\text{out}}(B_{\theta_D}(y) + G_{\theta_D}(y) \odot R_{\theta_D}(z, y)). \tag{4}$$

Here $B_{\theta_D}(y)$ supplies the observation-based prediction, $R_{\theta_D}(z, y)$ is the latent residual, and $G_{\theta_D}(y)$ is the gate, which controls how much the latent residual changes the base prediction. The product $\odot$ is elementwise. P_{out} maps it to the reconstructed field. We omit the mask and coordinates from the notation for brevity. Network and latent-branch details are given in Appendices A.1 and A.2.

Let $\mathcal{D}_p = \{(c, y^{\text{num}})\}$ be the small set of fields paired with their true PDE responses, and let $\mathcal{D}_u = \{c\}$ contain additional fields without numerical responses. The decoder condition $\bar{y}(c)$ follows the task's observation protocol: paired fields can be conditioned on either the numerical response or the predicted response $\widehat{y}$, while unpaired fields use $\widehat{y}$. The main autoencoder objective is

$$\mathcal{L}_{\text{AE}} = \lambda_y \mathcal{L}_{\text{obs}} + \lambda_c \mathcal{L}_{\text{rec}} + \lambda_{\text{phys}} \mathcal{L}_{\text{phys}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}. \tag{5}$$

The supervised response loss uses only $\mathcal{D}_p$; field reconstruction also uses $\mathcal{D}_u$:

$$\mathcal{L}_{\text{obs}} = \mathbb{E}_{(c, y^{\text{num}}) \sim \mathcal{D}_p} d_y(\widehat{y}, y^{\text{num}}),$$
$$\mathcal{L}_{\text{rec}} = \mathbb{E}_{c \sim \mathcal{D}_p \cup \mathcal{D}_u} \sum_{q \in \{\mu, z\}} \omega_q(c)\, d_c(D_{\theta_D}(q, \bar{y}(c)), c)\,.$$

Here d_y measures response error, while d_c is a field loss (squared error for continuous fields and binary cross-entropy for piece-wise constant fields). The physics loss is:

$$\mathcal{L}_{\text{phys}} = \mathbb{E}_{(c', y') \sim \mathcal{B}_{\text{phys}}} \left[\frac{1}{|\mathcal{W}|} \sum_{w \in \mathcal{W}} |r_{\text{PDE}}(c', y'; w)|^2 \right] + \mathcal{L}_{\text{IC/BC}}.$$

Here $r_{\text{PDE}}(c', y'; w)$ is the weak-form PDE residual obtained by testing the governing equation with $w \in \mathcal{W}$ and integrating by parts in space. The latent regularizer $\mathcal{L}_{\text{reg}}$ follows the VAE formulation (Kingma & Welling, 2014), pulling the encoder's latent distributions toward a standard Gaussian.

We evaluate weak PDE residuals on the encoder's forward predictions and the decoder's inverse reconstructions. Weak residuals are detailed in Appendix A.3; autoencoder training (Algorithm 1) and loss settings are in Appendix A.4 and the task-specific implementation sections.

3.2 Conditional latent Flow

After the autoencoder is trained, its encoder latent output supplies the flow target z. When the true observation is available, we use it to condition the Flow. Otherwise, we use the frozen encoder's predicted observation as the condition (Figure 1B).

We use the observation y to guide the evolution of the latent state z_τ through spatially varying scaling and shifting, inspired by FiLM (Perez et al., 2018), where $z_\tau = (1 - \tau)z_0 + \tau z$.

We train the conditional flow using the standard flow-matching objective (Lipman et al., 2023; Liu et al., 2022), with straight paths from Gaussian noise to the frozen encoder's latent codes and conditioning on the corresponding observations.

Flow training is summarized in Algorithm 2; architecture, objective, and task-specific conditioning are detailed in Appendices A.1 and A.5.

3.3 Inference

For inverse problem, given an observation y, we initialize the latent state with Gaussian noise z_0 and integrate $\dot{z}_\tau = v_{\theta_F}(z_\tau, \tau, y)$ from $\tau = 0$ to $\tau = 1$ to obtain a latent sample. The decoder uses this latent sample and y to generate the final reconstruction (Figure 1C).

Let $\Phi_{\theta_F}(y; z_0)$ denote the terminal latent sample. To reduce variability due to random sampling, we aggregate reconstructions from K independent Gaussian sources. Each sample is decoded as:

$$z^{(k)} = \Phi_{\theta_F}(y; z_0^{(k)}), \qquad c^{(k)} = D_{\theta_D}(z^{(k)}, y), \quad k = 1, \ldots, K. \tag{6}$$

The samples from Eq. (6) give alternative reconstructions for the observation. A point estimate is obtained using the aggregation rule specified for each experiment. Algorithm 3 summarizes inference.

Optional refinement. When observations are noisy, we can refine the response used to condition reconstruction. Each round updates this response using PDE and observation errors while keeping any clean-anchor values fixed, then reruns the frozen Flow and decoder to generate a new latent and reconstruct the unknown field. Only the response is gradient-optimized; network parameters remain fixed. The detailed procedure is given in Appendix A.6.

4 EXPERIMENTS

We study fine-detail recovery in two types of PDE inverse problems: recovering an initial field from a later state under Allen–Cahn dynamics, and recovering a Darcy coefficient from coarse or sparse pressure measurements. The Darcy experiments cover continuous coefficients and binary coefficients with sharp interfaces.

Baselines. We use parameter-matched No-Z controls that remove the complementary latent pathway from our AE and adjust decoder widths to match the parameter budget (Appendix B.1). PINO (Li et al., 2024) and DGenNO (Zang & Koutsourelakis, 2025) are evaluated on all three tasks; DGenNO is included only in the coarse-grid setting without anchors for continuous Darcy. For Allen–Cahn and continuous Darcy, we also compare with a conditional EDM (Karras et al., 2022) trained with a PDE-residual loss (Conditional EDM+PDE) evaluated through the frozen forward branch of our physics-aware encoder. Additional baselines are DPS (Chung et al., 2023) for Allen–Cahn and DiffusionPDE (Huang et al., 2024) for binary Darcy. All methods within each task use the same observations and test fields. Baseline settings are described in Appendix B.2 and Appendices C–E. DGenNO uses a latent size matching OCL-PDE (512 for Allen–Cahn and 1024 for Darcy). We perform direct latent optimization for DGenNO at test time; Appendix B.3 compares this procedure with the inference procedure used in the original paper.

Evaluation metrics. For continuous fields, we report relative L_2 error, $\|\widehat{c} - c\|_2/\|c\|_2$. For binary Darcy, the primary metric is hard pixel error: the fraction of grid points assigned the wrong coefficient value. Additional boundary-recovery metrics are introduced in Appendix B.4. We also assess forward consistency by independently solving the PDE with each reconstructed field and comparing the resulting solution with the reference solution using relative L_2 error. Test time is averaged per held-out field and excludes these independent numerical checks; for OCL-PDE, it includes generating and decoding all K samples used for the reconstruction.

4.1 BACKWARD RECOVERY FOR THE ALLEN–CAHN EQUATION

Inverse problem and observations. On the periodic domain $\Omega = [0, 2]^2$, the phase field $\phi(x, t)$ satisfies (Allen & Cahn, 1979)

$$\begin{aligned} \partial_t \phi(x,t) &= \nu \Delta_x \phi(x,t) + \phi(x,t) - \phi(x,t)^3, \quad && x \in \Omega,\ t > 0, \\ \phi(x,0) &= \phi_0(x), && x \in \Omega. \end{aligned} \tag{7}$$

Our goal is to recover the initial field ϕ_0 from a later full-field observation.

Experimental setup. With $\nu = 0.0004$, the observation is $\phi(\cdot, 4)$; both fields use a 128×128 grid. We sample $G \sim \mathcal{GP}(0, (-\Delta + I)^{-5/2})$ spectrally (Li et al., 2021) and normalize each initial field as $\phi_0 = G/\|G\|_\infty$. We evolve both true and recovered initial fields to $t = 4$ using Fourier spectral discretization and first-order semi-implicit time stepping (Shen & Yang, 2010), then measure the relative L_2 error between the resulting fields. The latent z has size $8 \times 8 \times 8$. At inference, we average $K = 64$ decoded samples. All methods use the same 500 supervised field–response pairs.

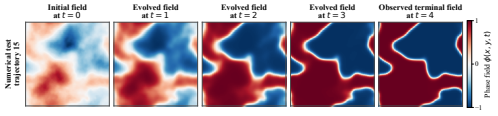


Figure 2: Allen–Cahn evolution of a trajectory from $t = 0$ to $t = 4$.

Implementation details are given in Appendices C.1–C.2. Fine details in the initial field are no longer resolved in the terminal observation, making backward recovery difficult as shown in Figure 2.

Table 1: Allen–Cahn initial-field and terminal-field relative L_2 errors on 20 held-out trajectories, with test time per field.

Method	ϕ_0 relative L_2 mean $\pm$ std.	Terminal-field relative L_2 at $t = 4$; mean $\pm$ std.	Test time (s/field)
PINO (Li et al., 2024)	0.18496 ± 0.04341	0.03110 ± 0.01316	–
DGenNO (Zang & Koutsourelakis, 2025) ($d = 512$)	0.66444 ± 0.33967	0.62601 ± 0.20541	73.51
Conditional EDM+PDE (Karras et al., 2022)	0.41914 ± 0.05692	0.41826 ± 0.07390	12.42
DPS (L2 guidance) (Chung et al., 2023)	0.45889 ± 0.14540	0.18284 ± 0.10136	27.56
No-Z ablation	0.09827 ± 0.01863	0.01679 ± 0.01108	–
OCL-PDE	$\mathbf{0.05897 \pm 0.01817}$	$\mathbf{0.01386 \pm 0.00547}$	0.9150

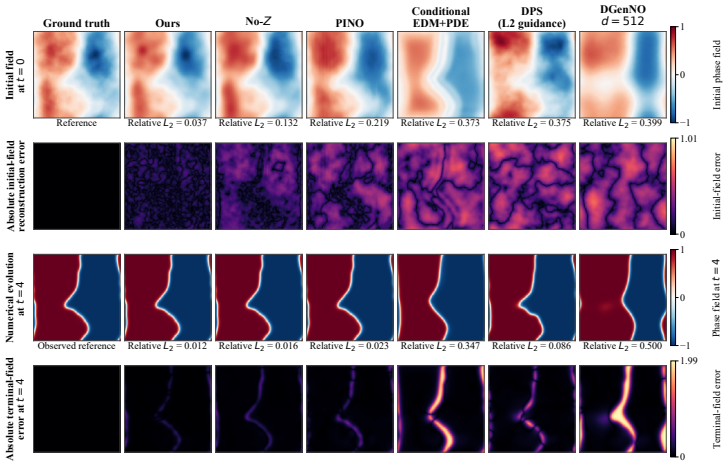


Figure 3: Allen–Cahn initial-field recovery. Rows show reconstructions, absolute initial-field errors, spectral-method evolutions to $t = 4$, and absolute terminal-field errors. We report initial-field and terminal-field errors in relative L_2.

We recover the Allen–Cahn initial field from its terminal observation $\phi(\cdot, t = 4)$ using OCL-PDE and other methods and compare both the reconstructed initial-field and its evolution with their respective ground truths. Initial-field relative L_2 error is lowest for OCL-PDE at 0.05897, compared with 0.18496 for PINO and 0.09827 for the parameter-matched No-Z control, which removes the latent pathway (Table 1). The reduction relative to No-Z is about 40%. Terminal-field error is also lowest at 0.01386. OCL-PDE better preserves fine-scale details within each phase, while most other methods mainly recover the overall phase layout (Figure 3). These results support the role of the complementary latent in recovering details unresolved by the terminal observation.

Test time is 0.9150 s per field for OCL-PDE, compared with 12.42 s for Conditional EDM+PDE, 27.56 s for DPS, and 73.51 s for DGenNO. OCL-PDE samples compact latents; Conditional EDM+PDE samples full fields, DPS and DGenNO repeatedly compute gradients. PINO and No-Z make direct predictions; their timings are omitted from this table. These measurements describe the evaluated inference procedures (Appendix C.3). Appendix C.5 separately evaluates the learned forward branch against reference trajectories.

4.2 Darcy flow with a continuous coefficient and coarse observations

Inverse problem and observations. For steady Darcy flow on $\Omega = [0, 1]^2$, the pressure $u(x)$ satisfies

$$-\nabla_x \cdot \big(a(x)\nabla_x u(x)\big) = f(x), \quad x \in \Omega, \qquad u(x) = 0, \quad x \in \partial\Omega. \tag{8}$$

We recover a spatially varying Darcy coefficient from coarse pressure observations and also test whether exact pressure measurements at selected locations improve recovery when the remaining observations are noisy.

Experimental setup. Here $f(x) \equiv 1$. We adapt the Gaussian random field generation and sample-wise normalization used in the Allen–Cahn example to the domain $[0,1]^2$. On the 128×128 grid, the resulting normalized field g defines the positive Darcy coefficient through $a = \exp(g)$. The continuous-coefficient Darcy experiment uses 1,000 paired fields. The latent z has size $4 \times 16 \times 16$. We average $K = 16$ decoded coefficient samples for the clean-anchor study. We solve for reference pressures and pressures from recovered coefficients using conservative finite differences with arithmetic face coefficients (Li et al., 2021; Huang et al., 2024), the relative L_2 error between these pressure fields is reportedin the u columns of Table 2.

We observe the pressure u on an aligned 32×32 grid and recover the coefficient a on a 128×128 grid. After removing the known zero boundary, the observation contains 900 interior $u(x)$ values. Implementation details and the training procedure are given in Appendix D.2.

We replace a subset of the noisy interior coarse-grid pressure values with noise-free reference values, called clean anchors. Writing $\mathcal{I}_H$ for the anchor set,

$$y_i^{\text{noisy}} = u_i + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, (\rho s_u)^2), \qquad y_i^{\text{anchor}} = u_i \text{ for } i \in \mathcal{I}_H, \qquad r = \frac{|\mathcal{I}_H|}{900}, \tag{9}$$

where u_i is the noise-free reference pressure at interior node i, s_u is the training-set pressure standard deviation, ρ is the relative noise level, and r is the anchor fraction among the interior nodes. We use $\rho \in \{0.01, 0.03\}$ and vary r from 0 to 0.5; the known zero boundary remains exact.

Pressure correction with clean anchors. We update non-anchor coarse pressure values using PDE and observation gradients, keeping clean anchors and the zero boundary fixed. The observation loss is normalized MSE against the fixed noisy measurements, penalizing excessive deviation from the data. Each update reconstructs the coefficient from the corrected pressure. We use 20 rounds without a forward solve (Algorithm 4). Appendix D.6 evaluates 0–50% clean-anchor coverage at both noise levels.

Table 2: GRF coefficient reconstruction on 16 test fields with 50% clean-anchor coverage: mean relative L_2 errors of the recovered coefficient (a) and corresponding pressure (u), before (r0) and after (r20) pressure correction. Pressure is computed from each recovered coefficient using conservative finite differences.

Method	Noise $\rho = 1\%$				Noise $\rho = 3\%$			
	a r0	a r20	u r0	u r20	a r0	a r20	u r0	u r20
PINO	0.03836	0.03669	0.00452	0.00450	0.05252	0.03694	0.00580	0.00456
Conditional EDM+PDE	0.09808	0.09912	0.02349	0.02398	0.09927	0.09893	0.02372	0.02400
No-Z ablation	0.03605	0.03151	0.00345	0.00318	**0.05127**	0.04034	0.00531	0.00409
OCL-PDE	**0.03456**	**0.02249**	**0.00310**	**0.00265**	0.05239	**0.02381**	**0.00520**	**0.00285**

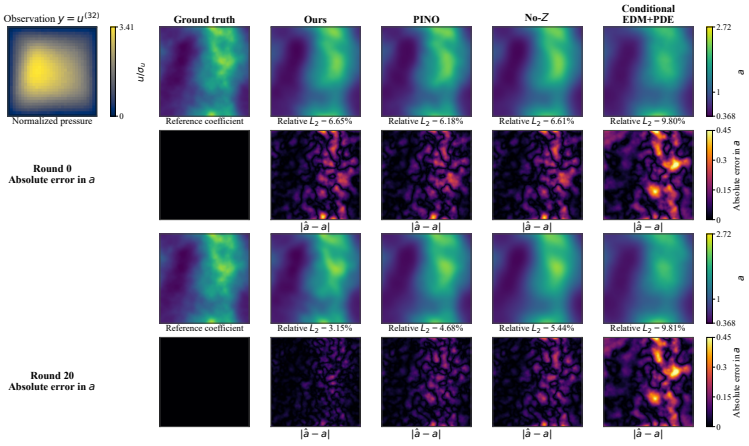


Figure 4: Continuous coefficient recovery before and after pressure correction at 3% noise and 50% clean-anchor coverage. We show the pressure observation, reference coefficient, reconstructions, and absolute coefficient errors, and report coefficient reconstruction errors in relative L_2.

We recover continuous Darcy coefficients from noisy coarse pressures with clean anchors and compare the methods before and after pressure correction. Coefficient error decreases from 0.03456 to 0.02249

at 1% noise and from 0.05239 to 0.02381 at 3% noise, reductions of 34.9% and 54.6%, respectively (Table 2). The largest baseline reduction at 3% noise is 29.7% (PINO). Fine coefficient variations are also recovered more faithfully after correction in the illustrated field (Figure 4). These results show that OCL-PDE makes more effective use of clean anchors to improve reconstruction accuracy, while the evaluated baselines benefit less from these high-quality observations.

Appendix D.7 tests noise robustness without clean anchors. Pressure error is lowest for OCL-PDE at every tested noise level. Coefficient relative L_2 error remains controlled, rising from 0.01807 without noise to 0.05740 at 3% noise, and is comparable to or below the evaluated baselines.

4.3 Darcy flow with a piecewise constant coefficient

Inverse problem and observations. The pressure follows Eq. (8), with constant forcing $f(x) \equiv 1$ and a two-valued coefficient $a(x) \in \{3, 12\}$. Our goal is to recover the piecewise constant coefficient from sparse pressure measurements.

Experimental setup. We sample $G \sim \mathcal{GP}(0, (-\Delta + 9I)^{-2})$ in a cosine basis without the constant mode and set $a(x) = 3 + 9\mathbf{1}_{\{G(x)\geq 0\}}$, following Darcy benchmarks (Li et al., 2021; Zang & Koutsourelakis, 2025). We recover a on a 128×128 grid from random interior pressure observations at 2%, 5%, and 10% observation rates. Training uses 1,000 paired fields, supplemented by dynamically generated coefficients during latent training. We encode the coefficient geometry as a truncated signed-distance field (TSDF) (Curless & Levoy, 1996) to construct a latent of size $4 \times 16 \times 16$ (Appendix E.1). We average $K = 16$ phase-probability samples and threshold at 0.5. The same conservative finite-difference solver computes pressures from the recovered hard coefficients, table 3's u columns report their relative L_2 errors against reference pressures. Implementation and training details are given in Appendix E.2.

Table 3: Binary coeficient reconstruction on 64 test fields: hard pixel error of the recovered coefficient, relative L_2 error of the corresponding pressure, and inference time (seconds per field). Pressure is computed from each recovered hard coefficient using conservative finite differences.

Method	2% (318 points)			5% (794 points)			10% (1588 points)		
	Hard error ↓	u rel. L_2 ↓	Time	Hard error ↓	u rel. L_2 ↓	Time	Hard error ↓	u rel. L_2 ↓	Time
PINO	0.06800	0.08264	-	0.05070	0.06317	-	0.04572	0.05456	-
DGenNO ($d = 1024$)	0.13957	0.10923	99.02	0.14084	0.11891	96.24	0.14459	0.12381	84.42
DiffusionPDE (Huang et al., 2024)	0.15472	0.06302	164.8	0.12350	0.05192	164.5	0.10744	0.04561	164.6
No-Z	0.04659	0.02709	-	0.03454	0.01950	-	0.03143	0.01753	-
OCL-PDE	**0.03524**	**0.01919**	0.3417	**0.02357**	**0.01199**	0.3437	**0.02105**	**0.01086**	0.3440

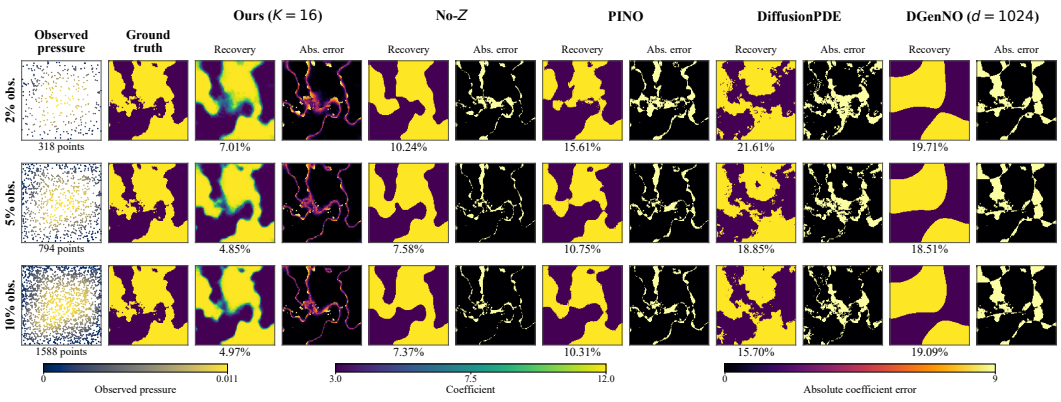


Figure 5: Binary coefficient recovery on the same held-out field at 2%, 5%, and 10% observation rates (top to bottom). Percentages below reconstructions are hard pixel errors.

We compare binary coefficient recovery and forward consistency at three observation densities. Hard pixel error and pressure relative L_2 error are lowest for OCL-PDE at every density (Table 3). At 5% observations, hard pixel error is 2.36%, compared with 5.07% for PINO, a 53.5% relative reduction. Error is also 31.8% lower than the No-Z ablation's 3.45%. Recovered region shapes and small holes

match the reference more closely (Figure 5), supporting the latent's role in recovering sharp interfaces from sparse observations.

Test time is approximately 0.34 s per field for OCL-PDE, 84–99 s for DGenNO, and 165 s for DiffusionPDE with 2,000 guided diffusion steps. Boundary F1 is highest and ASSD and HD95 are lowest at all densities, supporting improved interface detail recovery (Appendix E.3). Learned forward prediction is also most accurate (Appendix E.5). Without retraining or refinement, hard pixel error remains stable at 80 and 50 dB observation noise, and OCL-PDE outperforms No-Z and PINO at 2%, 5%, and 10% observation density (Appendix E.6).

4.4 Ablation study: what does the complementary latent contribute?

Does the latent improve reconstruction? OCL-PDE improves reconstruction over parameter-matched No-Z controls, which use only the observation pathway, across all three problems (Tables 1, 2, and 3). These gains demonstrate the benefit of the latent beyond the observation pathway alone.

Are y and z complementary? For Allen–Cahn, we exchange either the observation or the latent between two samples. The coarse arrangement and outlines of the two phase regions follow the observation: swapping y changes which phase occupies a given region. The spatial patterns within each phase follow the latent, changing when z is replaced while y remains fixed (Figure 6; further interventions in Appendix F.1). These swaps support the complementary roles of y and z: y primarily determines the phase arrangement, while z supplies fine-scale details within the phases.

What does the learned latent recover? We fix the observation and set $z = 0$ in the frozen decoder to obtain a base reconstruction. The latent correction follows spatial patterns missing from this base, partially recovering their locations and signs (Figure 7; Appendix Figures 10–12). Positive Q confirms that these corrections reduce the base's pooled squared error. Lower E_{hi} further shows improved agreement with the ground truth at high frequencies (Table 16, Appendix F.2). Thus, the latent restores missing spatial structure, including a portion of its fine-scale variation.

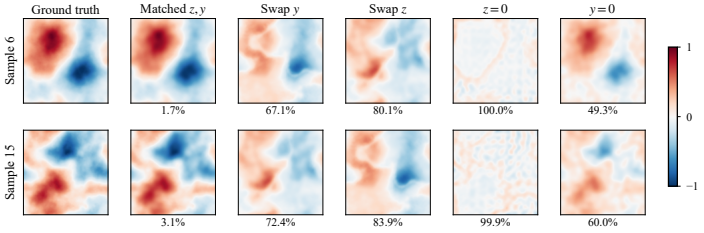


Figure 6: Allen–Cahn reconstructions for one pair of samples, swapping y or z between rows or setting either input to zero. Errors are relative L_2 against each row's ground truth.

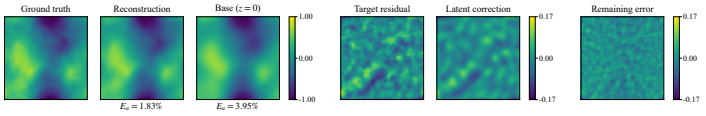


Figure 7: GRF Darcy latent correction in $\log a$ (Sample 0). The last three columns show what the base ($z = 0$) misses, what the latent adds, and what remains unrecovered, respectively. Coefficient errors are relative L_2.

5 Conclusion and Future Work

We presented OCL-PDE, which combines a physics-aware autoencoder with conditional Flow Matching to learn and generate observation-complementary latents for PDE inverse problems. Experiments on backward Allen–Cahn initial state recovery and continuous and binary Darcy coefficient inversion demonstrate improved reconstruction accuracy and detail recovery over the evaluated baselines. Ablations confirm that the latent supplies information beyond the observation, with task-dependent complementary roles.

Beyond initial-state recovery and coefficient inversion, future work will explore extending OCL-PDE to other PDE inverse problems, such as recovering heat-source distributions from temperature measurements and identifying unknown boundary conditions from interior measurements.

AI USE STATEMENT

We used large language models to assist with literature search, translation, manuscript revision, method implementation and debugging, experiment execution, and interpretation of experimental results. The authors reviewed the AI-assisted work and verified the references, implementation, and reported results. The authors take full responsibility for the final manuscript, implementation, and findings.

ETHICS STATEMENT

This work studies PDE inverse problems using synthetically generated physical fields and numerical simulations. It does not involve human participants, personal data, or animal experiments. We do not identify specific ethical concerns arising from the experiments reported here.

REPRODUCIBILITY STATEMENT

The appendix provides details of the network architectures, data generation, training settings, and evaluation protocols.

REFERENCES

Samuel M. Allen and John W. Cahn. A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening. *Acta Metallurgica*, 27(6):1085–1095, 1979. doi: 10.1016/0001-6160(79)90196-2. URL https://doi.org/10.1016/0001-6160(79)90196-2.

Gang Bao and Yaohua Zang. A unified physics-informed generative operator framework for general inverse problems, 2025. URL https://arxiv.org/abs/2511.03241.

Gang Bao, Xiaojing Ye, Yaohua Zang, and Haomin Zhou. Numerical solution of inverse problems by weak adversarial networks. *Inverse Problems*, 36(11):115003, 2020. doi: 10.1088/1361-6420/abb447.

Gang Bao, Chang Ma, and Yuxuan Gong. PFWNN: A deep learning method for solving forward and inverse problems of phase-field models. *Journal of Computational Physics*, 527:113799, 2025. doi: 10.1016/j.jcp.2025.113799. URL https://doi.org/10.1016/j.jcp.2025.113799.

Bowen Cheng, Ross Girshick, Piotr Dollár, Alexander C. Berg, and Alexander Kirillov. Boundary IoU: Improving object-centric image segmentation evaluation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15334–15342, 2021. doi: 10.1109/CVPR46437.2021.01508.

Hyungjin Chung, Jeongsol Kim, Michael Thompson McCann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*. OpenReview.net, 2023. URL https://openreview.net/forum?id=OnD9zGAGT0k.

Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 303–312. ACM Press, 1996. doi: 10.1145/237170.237269.

Weinan E and Bing Yu. The Deep Ritz method: A deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018. doi: 10.1007/s40304-018-0127-z.

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. doi: 10.1016/j.neunet.2017.12.012. URL https://doi.org/10.1016/j.neunet.2017.12.012.

Robert Eymard, Thierry Gallouët, and Raphaèle Herbin. Finite volume methods. In *Handbook of Numerical Analysis*, volume 7, pp. 713–1018. Elsevier, 2000. doi: 10.1016/s1570-8659(00)07005-8. URL https://doi.org/10.1016/s1570-8659(00)07005-8.

Bruce A. Finlayson. *The Method of Weighted Residuals and Variational Principles: With Application in Fluid Mechanics, Heat and Mass Transfer*, volume 87 of *Mathematics in Science and Engineering*. Academic Press, New York, 1972. URL https://ci.nii.ac.jp/ncid/BA048096 2X.

Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985. doi: 10.1016/0304-3975(85)90224-5. URL https://doi.org/10.1016/0304-3975(85)90224-5.

Bastian Harrach. An introduction to finite element methods for inverse coefficient problems in elliptic PDEs. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 123(3):183–210, 2021. doi: 10.1365/s13291-021-00236-2.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. IEEE, 2016. doi: 10.1109/cvpr.2016.90. URL https://doi.org/10.1109/cvpr.2016.90.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs), 2016. URL https://arxiv.org/abs/1606.08415.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

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.

Victor Isakov. *Inverse Problems for Partial Differential Equations*, volume 127 of *Applied Mathematical Sciences*. Springer International Publishing, 3 edition, 2017. doi: 10.1007/978-3-319-51658-5.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems 35*, pp. 26565–26577. Neural Information Processing Systems Foundation, Inc., 2022. doi: 10.52202/068431-1926. URL https://doi.org/10.52202/068431-1926.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations (ICLR)*, 2014. URL https://iclr.cc/archive/2014/conference-proceedings/.

I.E. Lagaris, A. Likas, and D.I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998. doi: 10.1109/72.712178. URL https://doi.org/10.1109/72.712178.

D. Lesnic, J. R. Berger, and P. A. Martin. A boundary element regularization method for the boundary determination in potential corrosion damage. *Inverse Problems in Engineering*, 10(2):163–182, 2002. doi: 10.1080/10682760290007453.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2021. URL https://arxiv.org/abs/2010.08895.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024. doi: 10.1145/3648506.

Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2117–2125, July 2017.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023. URL https://arxiv.org/abs/2210.02747.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow, 2022. URL https://arxiv.org/abs/2209.03003.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021. doi: 10.1038/s42256-021-00302-5.

Yulong Lu and Wuzhe Xu. Generative downscaling of PDE solvers with physics-guided diffusion models, 2024. URL https://arxiv.org/abs/2404.05009.

D.R. Martin, C.C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004. doi: 10.1109/tpami.2004.1273918. URL https://doi.org/10.1109/tpami.2004.1273918.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018. doi: 10.1609/aaai.v32i1.11671. URL https://ojs.aaai.org/index.php/AAAI/article/view/11671.

Maziar Raissi, Paris Perdikaris, and George 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.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*, pp. 234–241. Springer International Publishing, 2015. ISBN 9783319245744. doi: 10.1007/978-3-319-24574-4_28. URL https://doi.org/10.1007/978-3-319-24574-4_28.

Jie Shen and Xiaofeng Yang. Numerical approximations of Allen–Cahn and Cahn–Hilliard equations. *Discrete and Continuous Dynamical Systems*, 28(4):1669–1691, 2010. ISSN 1553-5231. doi: 10.3934/dcds.2010.28.1669. URL https://doi.org/10.3934/dcds.2010.28.1669.

Daniil Sherki, Ivan Oseledets, and Ekaterina Muravleva. Bayesian inverse problems meet flow matching: Efficient and flexible inference via transformers, 2025. URL https://arxiv.org/abs/2503.01375.

Dule Shu, Zijie Li, and Amir Barati Farimani. A physics-informed diffusion model for high-fidelity flow field reconstruction. *Journal of Computational Physics*, 478:111972, 2023. doi: 10.1016/j.jcp.2023.111972.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

A. M. Stuart. Inverse problems: A bayesian perspective. *Acta Numerica*, 19:451–559, 2010. doi: 10.1017/s0962492910000061. URL https://doi.org/10.1017/s0962492910000061.

Abdel Aziz Taha and Allan Hanbury. Metrics for evaluating 3d medical image segmentation: Analysis, selection, and tool. *BMC Medical Imaging*, 15(1):29, 2015. doi: 10.1186/s12880-015-0068-x.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40):eabi8605, 2021. doi: 10.1126/sciadv.abi8605.

Sifan Wang, Zehao Dou, Siming Shan, Tong-Rui Liu, and Lu Lu. FunDiff: Diffusion models over function spaces for physics-informed generative modeling. *Nature Communications*, 17(1):5749, 2026. doi: 10.1038/s41467-026-72292-0. URL https://www.nature.com/articles/s41467-026-72292-0.

Holger Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, 4(1):389–396, 1995. doi: 10.1007/BF02123482. URL https://doi.org/10.1007/BF02123482.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–19, 2018. URL https://openaccess.thecvf.com/content_ECCV_2018/html/Yuxin_Wu_Group_Normalization_ECCV_2018_paper.html.

Yaohua Zang and Phaedon-Stelios Koutsourelakis. Dgenno: a novel physics-aware neural operator for solving forward and inverse pde problems based on deep, generative probabilistic modeling. *Journal of Computational Physics*, 538:114137, 2025. doi: 10.1016/j.jcp.2025.114137. URL https://doi.org/10.1016/j.jcp.2025.114137.

Yaohua Zang, Gang Bao, Xiaojing Ye, and Haomin Zhou. Weak adversarial networks for high-dimensional partial differential equations. *Journal of Computational Physics*, 411:109409, 2020. doi: 10.1016/j.jcp.2020.109409.

A Architecture, training, and inference details

Throughout the appendix, *GRF Darcy* denotes the continuous-coefficient Darcy-flow experiment, in which the log-permeability field is generated from a Gaussian random field (GRF).

A.1 Shared network architecture

Physics-aware autoencoder. All three tasks use the same four-part design rather than a single CNN or a plain U-Net. A shallow convolutional stem is shared by an FNO-based deterministic response branch (Li et al., 2021) and a convolutional latent branch. The response branch predicts the observable or forward field, whereas the latent branch produces the compact complementary target z. A U-Net-style conditional decoder, inspired by the contracting/expanding organization and skip connections of U-Net (Ronneberger et al., 2015), reconstructs the unknown field from the predicted or measured response, its validity mask, coordinates, and z. The observation path produces both the base prediction $B_{\theta_D}(y)$ and a spatial gate $G_{\theta_D}(y)$. In parallel, the latent path projects z at multiple decoder scales and combines it with observation features to form the correction $R_{\theta_D}(z, y)$. As in Eq. (4), their outputs combine as

$$\widehat{c} = D_{\theta_D}(z, y) = P_{\text{out}}(B_{\theta_D}(y) + G_{\theta_D}(y) \odot R_{\theta_D}(z, y)) \,. \tag{10}$$

The base supplies the observation-based estimate, and the gate controls the latent correction at each location. The sum is formed before the task-specific output mapping P_{out}, which is the identity for Allen–Cahn; the Darcy mappings are specified in their implementation sections. The latent correction path is an addition to the original U-Net.

Conditional latent Flow. The velocity network v_{θ_F} takes the current latent state z_τ, Flow time τ, and an observation condition. Its spatial channels are $\eta_y = [\widetilde{y}, M_y, A_y]$, where $\widetilde{y}$ is the normalized observation (zero-filled where needed), M_y marks valid measurements, and A_y contains optional task-specific channels. A convolutional extractor C_{θ_F} maps η_y to features h_y at the latent resolution. An input convolution maps z_τ to the first hidden state h_0. In residual convolutional block l (He et al., 2016), a projection P_l of h_y produces a spatially varying scale and shift:

$$\begin{aligned} h_y &= C_{\theta_F}(\eta_y), \qquad (\gamma_y^{(l)}, \beta_y^{(l)}) = P_l(h_y), \\ \mathrm{Mod}_l(h_l; y) &= (1 + \gamma_y^{(l)}) \odot \mathrm{GN}(h_l) + \beta_y^{(l)}. \end{aligned} \tag{11}$$

Here h_l is the block's hidden state, GN is group normalization, and Mod_l is spatial FiLM-style conditioning (Perez et al., 2018). Flow time τ is embedded sinusoidally (Vaswani et al., 2017; Ho et al., 2020) and adds channel-wise modulation; Allen–Cahn also embeds its known observation time $t_{\mathrm{cond}} = 4$. One four-head self-attention block (Vaswani et al., 2017) separates two stacks of residual blocks. After the final block, group normalization, SiLU activation, and an output convolution produce $v_{\theta_F}(z_\tau, \tau, y)$ with the same shape as z_τ.

Allen–Cahn and Binary Darcy use multi-scale condition extractors inspired by the lateral projections of feature pyramid networks (Lin et al., 2017); GRF Darcy uses a single downsampling stage from its native observation grid. We call the multi-scale extractors *FPN-inspired* because they aggregate projected features at the latent resolution without the top-down pathway of the original FPN. Task-specific condition channels and Flow widths are given in Table 4 and the implementation sections below.

A.2 Latent construction

All tasks use direct full-field encoding. The latent branch receives a representation $h = T(c)$ of the complete unknown field: the initial phase field for Allen–Cahn, log-permeability for GRF Darcy, and a scaled signed-distance field for Binary Darcy. For both Darcy tasks, we center the latent mean at the zero input, $\mu = E^z_{\theta_E}(h) - E^z_{\theta_E}(0)$, where $E^z_{\theta_E}$ denotes the uncentered latent-mean branch. This fixes the latent reference at $h = 0$. The encoder supplies (μ, ℓ), and sampling follows Eq. (3). Task-specific transforms and latent sizes are given in Appendices C.1–E.1. The latent encodes the complete transformed field $T(c)$, not its difference from a zero-latent reconstruction. In every task, Flow is conditioned on the observation and the task-specific inputs listed in Table 4; $D(0, y)$ is not a Flow condition. The decoder's latent correction pathway and the zero-latent diagnostic remain distinct from this full-field encoding.

A.3 Weak-form physics losses

We construct weak-form PDE losses by the method of weighted residuals (Finlayson, 1972): multiply the governing equation by a test function w and integrate by parts in space. For example, applying this procedure to the Allen–Cahn and Darcy equations used in our experiments gives

$$r_{\mathrm{AC}}(\phi; w) = \int_\Omega \left[\partial_t \phi\, w + \nu \nabla\phi \cdot \nabla w + (\phi^3 - \phi) w\right]\, d\boldsymbol{x}, \tag{12}$$

$$r_{\mathrm{Darcy}}(a, u; w) = \int_\Omega \left[a \nabla u \cdot \nabla w - f w\right]\, d\boldsymbol{x}. \tag{13}$$

Here ϕ and u denote the forward solution y for Allen–Cahn and Darcy, respectively; $a = c$ is the Darcy coefficient, and f is its forcing. The compactly supported test functions below eliminate the spatial boundary term. These residuals require only first spatial derivatives of the response and no spatial derivatives of a.

For predicted responses in all three tasks, we use translated Wendland C^2 compactly supported radial basis functions (CSRBFs) (Wendland, 1995; Bao et al., 2025; Zang & Koutsourelakis, 2025),

$$w_{\boldsymbol{\xi}}(\boldsymbol{x}) = \psi\left(\frac{\|\boldsymbol{x} - \boldsymbol{\xi}\|_2}{R}\right), \qquad \psi(r) = (1 - r)_+^4 (4r + 1), \tag{14}$$

where $(r)_+ = \max(r, 0)$, $\boldsymbol{\xi}$ is a test center, and $R = 4h$ is the support radius, with h denoting the spatial sampling interval. Each $w_{\boldsymbol{\xi}} \in H_0^1(\Omega)$ vanishes outside the ball $B(\boldsymbol{\xi}, R)$, so its residual

can be evaluated using only a local neighborhood. The radius and discrete normalization here follow our training implementation. On the 128×128 training grids, we set $h = 2/127$ for Allen–Cahn and $h = 1/128$ for both Darcy tasks. With support radius $R = 4h$, each local weak residual is evaluated over a 9×9 neighborhood. We repeat this calculation at all grid points whose neighborhoods lie entirely within the domain. Spatial response derivatives are computed by second-order finite differences on the predicted fields. For Allen–Cahn, the temporal derivative is computed by automatic differentiation in the shared physical-time direction. Test-function derivatives are evaluated analytically. The CSRBF loss is the mean squared discrete residual over training samples and valid centers. Darcy uses quadrature-weighted sums with area factor h^2, whereas Allen–Cahn uses the unscaled local sums; these conventions determine the normalization of their physics-loss weights.

In both Darcy tasks, the CSRBF terms constrain predicted pressure fields with either supplied or reconstructed coefficients. During AE training, we also constrain reconstructed coefficients using fixed numerical pressure fields. This auxiliary weak PDE loss uses the same CSRBF test functions in Eq. (14). Pressure gradients are computed by centered finite differences. Each squared local residual is normalized by the squared local integral of $fw_{\boldsymbol{\xi}}$, with a lower bound of 10^{-20} on the denominator, and the loss averages over valid test centers and training samples. These are autoencoder-stage losses; the subsequent Flow stage uses only the velocity objective in Eq. (17).

A.4 AUTOENCODER TRAINING

Optimization stages. Autoencoder optimization follows task-specific schedules for the response branch, observation-based decoder path, and latent correction path (Appendices C–E).

Auxiliary variance head. The latent encoder produces the mean μ_z. An auxiliary 1×1 convolutional head predicts $\ell_z = H_{\text{var}}(\mu_z) = \log \sigma_z^2$ elementwise. Together they define the Gaussian distribution in Eq. (3). Both belong to θ_E in the main text; the "Log-var" column of Table 5 counts the head separately for parameter accounting.

Latent sampling and KL regularization. For all three PDE examples, initial Gaussian latent learning uses both the mean μ and sampled latents

$$z^\star = \mu + \alpha\sigma \odot \xi, \qquad \xi \sim \mathcal{N}(0, I). \tag{15}$$

Here $\odot$ denotes elementwise multiplication and α controls the sampling spread; $\alpha = 0$ gives the mean. During initial latent learning, α is annealed up to 0.5. The KL regularizer is averaged over latent components and training samples:

$$\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{KL}} = \mathbb{E}\left[\frac{1}{2d}\sum_{i=1}^{d}\left(\mu_i^2 + \sigma_i^2 - \log\sigma_i^2 - 1\right)\right]. \tag{16}$$

Its coefficient reaches 10^{-4} at full weight.

Autoencoder training algorithm. Algorithm 1 summarizes autoencoder training. Task-specific observation protocols, loss weights, and schedules are given in Appendices C–E.

Algorithm 1 Autoencoder training

Require: Training fields c and paired numerical responses when available
Ensure: Encoder E_{θ_E} and decoder D_{θ_D}
1: **for** each autoencoder update **do**
2: Sample fields and available paired responses; obtain $\widehat{y}$ from E_{θ_E} with input c.
3: Form y from predicted or numerical responses using the observation protocol.
4: Encode the complete field c to obtain (μ, ℓ) using Appendix A.2; set $\sigma = \exp(\ell/2)$.
5: Form $z^\star = \mu + \alpha\sigma \odot \xi$ with $\xi \sim \mathcal{N}(0, I)$.
6: Reconstruct c from both (μ, y) and $(z^\star, y)$ using D_{θ_D}.
7: Update the scheduled subsets of (θ_E, θ_D) using Eq. (5).
Evaluate $\mathcal{L}_{\text{obs}}$ only where paired responses are available.
8: **end for**

A.5 Conditional Flow training and conditioning

The autoencoder is frozen throughout Flow training.

Flow targets and normalization. Flow targets use $\alpha = 0.5$ for Allen–Cahn and Binary Darcy, and $\alpha = 0$ for both GRF observation settings, so GRF targets are encoder means. This does not remove the Gaussian source used for Flow inference. Allen–Cahn uses source standard deviation $\sigma_0 = 0.04$ and no latent standardization. Darcy uses $\sigma_0 = 1$ and standardizes target latents with per-channel means and standard deviations estimated from training data; this transform is inverted before decoding. These fixed transforms are included in Φ_{θ_F} in the main text. Allen–Cahn additionally pairs source and target samples by minibatch optimal transport before velocity matching.

Flow objective. We train the Flow with straight conditional Flow Matching (equivalently, the one-step rectified-flow objective) (Lipman et al., 2023; Liu et al., 2022):

$$z_\tau = (1-\tau)z_0 + \tau z_1, \qquad v^\star = z_1 - z_0, \qquad \mathcal{L}_{\mathrm{FM}} = \mathbb{E}\big[\|v_{\theta_F}(z_\tau, \tau, y) - v^\star\|_2^2\big]. \tag{17}$$

"One-step" here denotes one rectification stage in the learning objective, not one numerical integration step: we do not perform iterative reflow, and all formal Flow evaluations below use 50 Euler steps.

Flow training algorithm. Algorithm 2 summarizes Flow training. Task-specific observation protocols and schedules are given in Appendices C–E.

Algorithm 2 Conditional Flow training

Require: Frozen trained encoder E_{θ_E}, training fields c, and paired numerical responses when available
Ensure: Velocity network v_{θ_F}

1: **for** each Flow update **do**
2: Form (μ, ℓ, y) using the frozen autoencoder and observation protocol.
3: Form $z^\star$ using the stated sampling scale and convert it to the Flow target z_1.
4: Draw $z_0 \sim \mathcal{N}(0, \sigma_0^2 I)$ and $\tau \sim \mathcal{U}[0,1]$.
5: Apply the stated source–target pairing and set $z_\tau = (1-\tau)z_0 + \tau z_1$.
6: Update θ_F using Eq. (17).
7: **end for**

Conditioning channels. The condition extractor C is part of the Flow velocity network v_{θ_F}. It converts the observation channels into spatial features on the latent grid. The velocity network uses these features, the current latent z_τ, and Flow time τ to predict a velocity with the same shape as z_τ. The task-specific observation channels are collected in Table 4 and explained in the implementation sections below. We write $\eta(y)$ for these channels to distinguish them from the unknown physical field c.

Task	Spatial condition supplied to C	Latent/velocity shape	Scalar inputs
Allen–Cahn	$\eta_{\mathrm{AC}} = [\phi_4, \mathbf{1}]$ on 128^2	$8 \times 8 \times 8$	τ and $t_{\mathrm{cond}} = 4$
GRF Darcy, no anchors	$\eta_{\mathrm{low}} = [u/s_u, M]$ on 32^2	$4 \times 16 \times 16$	τ only
GRF Darcy, clean anchors	$\eta_{\mathrm{MF}} = [u/s_u, M, M_H]$ on 32^2	$4 \times 16 \times 16$	τ only
Binary Darcy	$\eta_{\mathrm{bin}} = [y_q/s_u, M_q]$ on 128^2	$4 \times 16 \times 16$	τ only

Table 4: Spatial inputs to the Flow condition extractor C. Here ϕ is the Allen–Cahn solution and $\phi_4 = \phi(\boldsymbol{x}, 4)$ is its observed terminal field (y); $\mathbf{1}$ is the all-one validity mask. For GRF Darcy, u denotes the native 32^2 pressure observation; for Binary Darcy, y_q is the raw zero-filled 128^2 pressure tensor. The quantity s_u is the training-set pressure standard deviation. In the clean-anchor row, selected noisy values in u have been replaced with clean anchors. The masks M and M_q mark observed pressure nodes, and M_H marks clean anchors. In GRF Darcy, Flow receives native-grid observations. Flow time τ and physical time t_{cond} are separate scalar inputs; the noise level ρ is not supplied to any Flow.

A.6 Inference and optional refinement

Forward prediction uses the response output $\widehat{y}$ of $E_{\theta_E}(c)$. Algorithm 3 describes inverse sampling: the frozen Flow generates latents conditioned on the observation, and the frozen decoder reconstructs the unknown field. Optional refinement updates the response used as the condition under PDE and

observation constraints, while keeping trusted measurements and network weights fixed. Reconstruction is then repeated with the same Gaussian source samples. The continuous-Darcy refinement procedure is specified in Appendix D.3.

Algorithm 3 Inverse sampling and optional refinement

Require: Observation y, frozen D_{θ_D} and v_{θ_F}, and sample count K
Ensure: Reconstructed fields and an optional point estimate
1: **for** $k = 1, \ldots, K$ **do**
2: Draw $z_0^{(k)} \sim \mathcal{N}(0, \sigma_0^2 I)$ independently.
3: Integrate $\dot{z}_\tau = v_{\theta_F}(z_\tau, \tau, y)$ from $\tau = 0$ to 1.
4: Undo any latent normalization to obtain $z^{(k)} = \Phi_{\theta_F}(y; z_0^{(k)})$.
5: Decode $c^{(k)} = D_{\theta_D}(z^{(k)}, y)$.
6: **end for**
7: If needed, form a point estimate using the specified aggregation rule.
8: If refinement is used, update the condition and repeat reconstruction with fixed sources, following Algorithm 4.
9: **return** the reconstructed fields and the optional point estimate.

B Experiment Settings

B.1 Model configurations and parameter counts

Parameter counts. Frozen components remain loaded during inference. Accordingly, the totals in Table 5 count all parameters loaded by the audited inference graph, not parameters trained simultaneously in one optimization stage.

For Allen–Cahn, PINO's forward and inverse networks total 52.74M parameters ($+0.02\%$ relative to OCL-PDE's 52.73M). DPS uses 41,699,617 prior parameters and 11,167,993 forward-operator parameters, totaling 52,867,610 (52.87M), a $+0.26\%$ difference from OCL-PDE.

For GRF Darcy, PINO and the No-Z clean-anchor model have 40.58M and 40.70M parameters, respectively (-0.37% and -0.09% relative to OCL-PDE's 40.73M). These differences use the recorded components for each procedure; no uniform matching tolerance is claimed for all methods.

For Binary Darcy, the audited PINO, No-Z, DGenNO ($d = 1024$), and DiffusionPDE models contain 40.58M, 40.91M, 40.75M, and 41.92M parameters, respectively, within -0.58% to $+2.71\%$ of OCL-PDE's 40.82M.

Task	Field grid	Latent $C \times H \times W$	AE	Log-var	Flow	Total
Allen–Cahn	128×128	$8 \times 8 \times 8$	30,650,596	72	22,080,456	52,731,124
GRF Darcy	128×128	$4 \times 16 \times 16$	30,313,272	20	10,417,444	40,730,716
Binary Darcy	128×128	$4 \times 16 \times 16$	30,313,272	20	10,503,668	40,816,960

Table 5: Audited parameter counts for the models used in formal inference. The GRF Flow with clean anchors has one additional clean-anchor-mask condition channel.

B.2 Baseline architectures and PDE losses

PINO forward and inverse networks. Across Allen–Cahn, continuous Darcy, and binary Darcy, each PINO baseline consists of two separate FNO networks: a forward network $F_{\mathrm{PINO}} : c \mapsto y$ and an inverse network $G_{\mathrm{PINO}} : y \mapsto c$. For Darcy these maps are $a \mapsto u$ and $u_{\mathrm{obs}} \mapsto a$; for Allen–Cahn they map the initial field to a later field and the observed later field back to the initial field. Both networks belong to the PINO baseline and are trained independently of the OCL-PDE autoencoder. All network weights are frozen at evaluation.

For continuous Darcy, the inverse inputs are normalized observed pressure, an observation mask, forcing, and the two spatial coordinates. With clean anchors, the mask is replaced by the validity-plus-anchor status channel. The forward FNO is frozen during inverse training. Its pressure prediction initializes the first clean-anchor refinement step; later steps retain the accepted pressure iterate.

Baseline PDE losses. PINO uses the same CSRBF weak-residual formulation as OCL-PDE for Allen–Cahn and both Darcy tasks, including GRF settings with and without clean anchors. In the shared grid-based implementation, second-order finite differences evaluate spatial field derivatives inside the weak residual; Allen–Cahn time derivatives use automatic differentiation in the physical-time direction. Conditional EDM+PDE uses a CSRBF loss on the predicted response: pressure for Darcy and the learned forward response for Allen–Cahn. AC DPS retains its unconditional prior and uses the retrained forward operator for observation and CSRBF PDE guidance. Binary DiffusionPDE uses a grid finite-difference PDE residual during inference. DGenNO uses CSRBF weak residuals with response derivatives evaluated at independent query points. Task-specific training and inference settings are given in Appendices C–E.

B.3 DGenNO inverse inference protocols

Direct-latent inference used in our comparisons. We freeze the trained DGenNO generators and optimize a separate latent variable β for each test field. We use eight initializations: the training-code mean, observation-matched training codes when a code bank is available, and uniformly sampled codes to fill the remaining starts. Matching uses only the available response observations and their mask. For each initialization, we run 500 Adam steps on latent coordinates scaled by their training-code standard deviations, with projection to the latent bounds, followed by at most 50 L-BFGS iterations.

The objective combines observation mismatch and a weak PDE-residual loss using 32 residual centers; Allen–Cahn additionally includes initial-condition consistency between the generated initial field and response branch. We retain the lowest-objective candidate encountered throughout all iterations and restarts, without using the test field's ground-truth reconstruction error for selection. The resulting single β^* is decoded into the reconstructed coefficient or initial field.

This direct-latent multistart procedure differs from the original Beta-MLP parameterization, which also optimizes the latent representation at test time. In Binary Darcy, we threshold the selected generator's probability at $p \geq 0.5$ to obtain the hard coefficient field used for visualization and hard pixel error.

Original Beta-MLP inference procedure. We separately evaluate the original method's Beta-MLP parameterization on our checkpoints and test fixtures. The network has three ReLU hidden layers of widths 64, 64, and 128, followed by a d-dimensional Tanh output. One fit uses 1,000 Adam steps with weight decay 10^{-4}, learning rate 10^{-2} divided by three every 250 steps, 100 weak-residual centers, and 25 integration points. We set $\lambda_{\mathrm{PDE}} = 1$ and $\lambda_{\mathrm{data}} = 50$ for Binary Darcy or 25 for continuous fields. The implementation returns a single optimized β^*. This is a paper-aligned adaptation, not an exact reproduction: checkpoints, coefficient values, observation geometries, and test fixtures differ from the original experiments. Allen–Cahn is a task adaptation outside the original paper. All times below exclude independent numerical verification; entries are mean $\pm$ sample standard deviation. The task-specific appendix tables report this paper-aligned Beta-MLP inference using the same frozen checkpoints as the direct-latent evaluations.

B.4 Additional Metrics for Binary Darcy

Results using these definitions are reported in Table 3 for hard pixel error and solver-consistency error, and in Appendix E.3, Table 13, for Boundary F1, ASSD, and HD95 at 2%, 5%, and 10% observation rates.

Tolerance-based boundary F-measures compare contour precision and recall while allowing small localization errors (Martin et al., 2004; Cheng et al., 2021). Nearest-surface distance summaries are likewise standard for segmentation evaluation, and a Hausdorff quantile reduces sensitivity to isolated outliers (Taha & Hanbury, 2015). We specify our exact pixel-grid conventions below because symmetrization, tolerance, and empty-set handling vary across implementations.

The Binary-Darcy metrics are computed per held-out field on the $H \times W = 128 \times 128$ grid, then averaged arithmetically across fields. Let S denote the true binary phase mask: at pixel p, $S(p) = 1$ if $a(p) = 12$ and $S(p) = 0$ if $a(p) = 3$. The predicted mask $\widehat{S}$ is defined analogously from the reconstructed hard coefficient $\widehat{a}(p) \in \{3, 12\}$. Hard pixel error (HardErr) is the fraction

of all grid pixels whose predicted coefficient value differs from the true value. The independent solver-consistency error is

$$\mathrm{SolverErr} = \frac{\|\mathcal{F}(\widehat{a}) - \mathcal{F}(a)\|_2}{\|\mathcal{F}(a)\|_2}, \tag{18}$$

where $\mathcal{F}$ is an independent numerical Darcy solver using the benchmark forcing and boundary conditions. SolverErr measures the relative pressure error induced by the reconstructed hard coefficient.

For any binary mask S, its boundary-pixel set is

$$B(S) = \left\{ p : \max_{q \in \mathcal{N}_3(p)} S(q) \neq \min_{q \in \mathcal{N}_3(p)} S(q) \right\}, \tag{19}$$

where $\mathcal{N}_3(p)$ is the centered 3×3 pixel neighborhood. For Binary Darcy, p is a boundary pixel if this neighborhood contains both phases, i.e., pixels with coefficient values $a = 3$ and $a = 12$; the same rule applies to the predicted coefficient $\widehat{a}$. Thus $B(S)$ is a narrow band around the two-phase interface, whereas S labels the entire regions.

Let $\mathcal{D}_\delta(B)$ be the set of grid pixels whose absolute row and column offsets from at least one boundary pixel in B are both at most δ. For the reported tolerance $\delta = 1$, $\mathcal{D}_1(B)$ is the union of the centered 3×3 neighborhoods of all pixels in B, including B itself. A predicted boundary pixel is counted as correctly localized if at least one true boundary pixel lies within one row and one column of it; equivalently, it lies in $\mathcal{D}_1(B(S))$. This rule tolerates a small location error without requiring exact overlap or one-to-one pairing of boundary pixels. Boundary F1 is then

$$P_1 = \frac{|B(\widehat{S}) \cap \mathcal{D}_1(B(S))|}{\max(|B(\widehat{S})|, 1)}, \qquad R_1 = \frac{|B(S) \cap \mathcal{D}_1(B(\widehat{S}))|}{\max(|B(S)|, 1)}, \qquad \mathrm{BF1}_1 = \frac{2P_1R_1}{\max(P_1 + R_1, 10^{-8})}. \tag{20}$$

Here P_1 is the fraction of predicted boundary pixels close to a true boundary, and R_1 is the fraction of true boundary pixels covered by a nearby predicted boundary. "Close" allows a one-pixel offset horizontally, vertically, or diagonally. Extra predicted edges reduce precision, while missed true edges reduce recall; their harmonic mean rewards agreement in both directions. Boundary F1 ranges from zero to one, with higher values indicating better boundary agreement, not a higher fraction of correctly classified region pixels. The maxima in the denominators avoid division by zero.

For the distance metrics, let $d(p, B) = \min_{q \in B} \|p - q\|_2$ in pixel units and form the bidirectional nearest-boundary distance vector

$$\Delta(\widehat{S}, S) = \big(d(p, B(S))\big)_{p \in B(\widehat{S})} \,\|\, \big(d(q, B(\widehat{S}))\big)_{q \in B(S)}. \tag{21}$$

The first vector contains one distance per predicted boundary pixel to its closest true boundary pixel; the second contains one distance per true boundary pixel to its closest predicted boundary pixel. Here $\|$ denotes concatenation: the two vectors are placed end to end, not maximized or added elementwise, and repeated distances are retained. For example, $(0, 1, 1) \,\|\, (0, 2) = (0, 1, 1, 0, 2)$. A horizontal or vertical one-pixel offset has distance 1, while a diagonal one-pixel offset has distance $\sqrt{2}$. We summarize the combined vector as

$$\mathrm{ASSD} = \mathrm{mean}(\Delta), \qquad \mathrm{HD95} = Q_{0.95}(\Delta). \tag{22}$$

ASSD measures the average boundary displacement, giving equal weight to every entry in the combined vector. HD95 summarizes larger local boundary mismatches: approximately 95% of the distances fall at or below this value. Using this percentile rather than the single largest distance reduces the influence of a few isolated extreme errors. Both metrics are measured in pixels, with lower values indicating better alignment. As a hypothetical single-field example, ASSD $= 4$ px means that the bidirectional nearest-boundary distances average 4 pixels, not that every boundary pixel is displaced by 4 pixels. HD95 $= 4$ px instead means that approximately 95% of these distances are at most 4 pixels; the remaining tail may be larger. These metrics are calculated for each field before averaging across fields; the reported HD95 is therefore the mean of field-wise percentiles, not a percentile pooled over all fields. If either boundary set is empty, both distances are assigned the finite worst-case image diagonal $\sqrt{H^2 + W^2}$ pixels rather than infinity or a missing value.

C Allen–Cahn implementation and additional results

C.1 Latent construction

For Allen–Cahn, the unknown field is the initial state $c = \phi_0$. The latent branch encodes this initial field directly:

$$(\widehat{\phi}_t, \mu_z, \ell_z) = E_{\theta_E}(\phi_0), \qquad \mu_z \in \mathbb{R}^{8\times8\times8}. \tag{23}$$

The latent computation uses the encoder's shared convolutional stem. The latent contains eight channels on an 8×8 grid. It is learned jointly with a decoder that also receives the terminal observation, encouraging it to retain information useful for initial-field reconstruction.

C.2 Data, networks, and training

Initial-field generation. On the periodic domain $[0, 2]^2$, we generate $G \sim \mathcal{GP}(0, (-\Delta+I)^{-5/2})$ by spectral sampling with the zero Fourier mode removed, and normalize each sample as $\phi_0 = G/\|G\|_\infty$. This follows the spectral Gaussian-process construction used in neural-operator benchmarks (Li et al., 2021), with the covariance and normalization specified here.

Network architecture. Both the initial field and terminal observation are represented on a 128×128 grid, and the latent tensor has shape $8 \times 8 \times 8$. The forward branch contains four Fourier blocks of width 104 with 8×8 retained modes. Each block adds a spectral convolution to a pointwise 1×1 convolution and applies GroupNorm (Wu & He, 2018) and SiLU (Elfwing et al., 2018); the output head uses a $\tanh$ activation. The latent CNN contains three shallow convolutional/residual feature stages followed by four stride-2 downsampling blocks.

The decoder has four downsampling stages, a bottleneck, and four upsampling stages with skip concatenations. Its channel schedule is $64/96/128/192/256$. The parallel latent correction path injects projected z features at full, $1/2$, $1/4$, $1/8$, and bottleneck resolutions before the gated merge with the base reconstruction. Decoder blocks use GroupNorm, SiLU, reflection padding, and bilinear upsampling. The Flow condition module is a four-scale FPN-inspired extractor with lateral 1×1 projections; the projected scales are pooled and fused at the 8×8 latent resolution, without a top-down pathway. The Flow backbone has width 256, five spatially modulated residual blocks before and after one four-head self-attention block.

Encoder, decoder, and Flow inputs. Let B denote the batch size. The AE receives the initial field $\phi_0 \in \mathbb{R}^{B\times1\times128\times128}$ and the query map $[x_1, x_2, t] \in \mathbb{R}^{B\times3\times128\times128}$. Its response branch returns $\widehat{\phi}_t \in \mathbb{R}^{B\times1\times128\times128}$, and its latent branch returns a mean of shape $B \times 8 \times 8 \times 8$. The decoder receives z, the terminal field (predicted ϕ_t during AE training or observed ϕ_4 at inference), an all-one validity mask, and $[x_1, x_2, t = 4]$, and reconstructs $\phi_0 \in \mathbb{R}^{B\times1\times128\times128}$. The Flow maps $z_\tau \in \mathbb{R}^{B\times8\times8\times8}$ and the two-channel field $[\phi_4, \mathbf{1}] \in \mathbb{R}^{B\times2\times128\times128}$, together with $\tau \in \mathbb{R}^{B\times1}$ and $t_{\text{cond}} = 4 \in \mathbb{R}^{B\times1}$, to a velocity of the same shape as z_τ. Parent-Flow training uses the frozen AE prediction of ϕ_4 from each fresh GRF draw; calibration and formal testing use the raw numerical ϕ_4. The mask remains all ones. No training-standard- deviation normalization is applied because the field is already approximately in $[-1, 1]$. Throughout, the formal condition is $t = 4$ and ϕ_4.

Training and evaluation settings. The AE is optimized for 50,000 updates with AdamW (Loshchilov & Hutter, 2019) at peak learning rate 3×10^{-4}. The parent Flow is optimized for 30,000 updates at 10^{-4}. Formal numerical-ϕ_4 calibration then updates only the 195,264 parameters of the Flow condition encoder for 200 updates at 10^{-5}; the remaining Flow parameters stay frozen. These stages use weight decay 10^{-4} and cosine decay (Loshchilov & Hutter, 2017) to 10^{-6}. Training uses seed 0.

The AE uses 500 numerical field–response pairs across observation times and a fresh online GRF draw at every update. With the AE frozen, the parent Flow constructs its training pairs dynamically from fresh GRFs. Calibration reuses 100 of these numerical ϕ_4 conditions with an 80/20 split generated using seed 20260715. The formal test uses the 20 fixed held-out trajectories, $K = 64$ Flow samples, 50 Euler steps, and stochastic-sampling seed 9091.

C.3 ITERATIVE INFERENCE AND TIMING

Conditional EDM+PDE trains for 20 epochs using only the denoising loss, then linearly increases λ_{PDE} from zero to 100 over the next 20 epochs and holds it fixed thereafter. The objective is $10\mathcal{L}_{\mathrm{EDM}}+64\lambda_{\mathrm{PDE}}\mathcal{L}_{\mathrm{PDE}}$. The PDE residual uses the frozen forward operator, spatial field differences, and a physical-time Jacobian–vector product. TF32 is disabled.

For the AC comparison, Conditional EDM+PDE uses the final raw weights after 110 training epochs, $K = 64$ samples, and 25 Karras–Heun steps. The sampler evaluates the denoiser twice per step except at the final step, giving 49 evaluations per sample; it runs without test-time gradients.

DPS uses one chain with 1,000 reverse-diffusion steps per field. Its L2 observation loss is backpropagated through the denoiser and learned forward operator. The weak PDE term has weight 1 and is enabled during the last 20% of the chain, when the observation term is weighted by 0.1. The sampler applies a guidance step size of 0.01 to the combined gradient. DPS uses the retained unconditional prior and the retrained forward operator. Its mean sampling time is 27.56 s per field, excluding the independent numerical rollout used to compute FDM error.

DGenNO uses eight starts, 500 Adam steps per start, and up to 50 L-BFGS iterations.

OCL-PDE uses $K = 64$ latent samples and 50 Euler steps, followed by decoding and averaging. The reported times are recorded inference costs for these procedures, rather than a controlled comparison with identical sample counts and timer boundaries.

C.4 DGENNO WITH THE ORIGINAL INFERENCE PROCEDURE

We adapt DGenNO's original inverse inference procedure to recover Allen–Cahn initial fields. The original paper (Zang & Koutsourelakis, 2025) reports no Allen–Cahn experiment, so our task-specific training provides a reference comparison (Table 6).

Table 6: Paper-aligned Allen–Cahn task adaptation ($N = 20$). Errors are relative L_2. The discrepancy between learned and independently solved terminal responses indicates a forward-surrogate limitation.

Method	d	Initial ϕ_0	Learned response	Solver response	Time (s)
DGenNO	512	1.57088 ± 0.51192	0.36875 ± 0.20238	0.67583 ± 0.33643	16.67510 ± 3.37500

C.5 FORWARD PREDICTION

Our learned response operator $E_y(\phi_0, t)$ takes time as an input, so one checkpoint represents the continuous-time forward map $\phi_0 \mapsto \phi_t$. We evaluate the retrained response operator at $t \in \{1, 2, 3, 4\}$ using mean per-trajectory relative L_2 error over the same 20 held-out trajectories. EDM and DPS define no learned forward operator. OCL-PDE has the lowest forward relative L_2 error at all four evaluated times.

Method	Relative L_2 error ↓			
	$t = 1$	$t = 2$	$t = 3$	$t = 4$
PINO	0.011347	0.013178	0.017165	0.02221
DGenNO ($d = 512$)	0.36088	0.41163	0.45939	0.49121
No-Z ablation	0.010124	0.010537	0.011881	0.014101
OCL-PDE	0.009851	0.009926	0.011308	0.013504

D GRF-COEFFICIENT DARCY IMPLEMENTATION AND ADDITIONAL RESULTS

D.1 LATENT CONSTRUCTION

For continuous Darcy, the unknown field is the permeability $c = a$, and the networks represent it as $g = \log a$. The latent branch directly encodes g and centers its mean using the zero field:

$$\mu_z = E^z_{\theta_E}(g) - E^z_{\theta_E}(0), \qquad \mu_z \in \mathbb{R}^{4\times 16\times 16}. \tag{24}$$

The decoder retains the observation-conditioned base $g_0(y)$. The same latent construction is used with and without clean anchors.

D.2 DATA, NETWORKS, AND TRAINING

Network architecture. The permeability and pressure fields use 128×128 grids and the latent tensor has shape $4 \times 16 \times 16$. The deterministic Darcy branch has five Fourier blocks of width 80 with 10×10 modes. Each block combines a spectral and pointwise convolution, uses GELU (Hendrycks & Gimpel, 2016) between blocks, and uses SiLU in the projection head. A parameter-free analytic Dirichlet envelope multiplies the output to impose the zero pressure boundary exactly (Lagaris et al., 1998). The latent CNN has three convolutional/residual stages and three stride-2 downsampling blocks. The four-level U-Net-style decoder uses channels $64/96/128/192/256$ and a parallel five-scale latent correction path, with GroupNorm, SiLU, reflection padding, and bilinear upsampling.

For both coarse-grid $M = 32$ observation settings, the Flow condition extractor uses one stride-2 stage and a 1×1 projection to produce features at the 16×16 latent resolution. The Flow backbone has width 192, four spatially modulated residual blocks, one four-head self-attention block, and four further residual blocks.

Encoder and decoder inputs. The hidden representation is $g = \log a$. The response operator receives $g \in \mathbb{R}^{B\times1\times128\times128}$, coordinates $[x_1, x_2] \in \mathbb{R}^{B\times2\times128\times128}$, and forcing $f \in \mathbb{R}^{B\times1\times128\times128}$, and outputs the full pressure field in $\mathbb{R}^{B\times1\times128\times128}$. The latent mean has shape $B \times 4 \times 16 \times 16$. Given z, condition pressure, validity mask, coordinates, and forcing, the decoder returns raw g logits on 128^2; the physical postprocessing is $g = \tanh(D(\cdot))$ and $a = \exp(g)$.

Flow conditioning. The Flow receives native 32×32 pressure observations, their validity mask M, and, when applicable, a clean-anchor mask M_H. Pressure is divided by the training-set standard deviation s_u. Without clean anchors,

$$\eta_{\text{low}} = [u_{32}/s_u, M], \tag{25}$$

while replacing selected noisy values with exact measurements gives

$$\eta_{\text{MF}} = [u_{\text{mix}}/s_u, M, M_H]. \tag{26}$$

Both conditions stay on the native 32^2 grid and contain no noise-level token. The decoder retains its 128^2 inputs, obtained by endpoint-aligned nodal bilinear interpolation of pressure with a dense interior validity mask. The Flow is trained with anchor fractions from 0–50% and encoder-mean targets; inference starts from Gaussian latent sources.

Training and evaluation settings. AE training is sequential: we first train the forward branch for 100 epochs, then the observation-conditioned base decoder with $z = 0$ for 30 epochs, and finally introduce z to reconstruct the full coefficient field for 80 epochs. Each epoch has 100 optimizer updates, giving 21,000 updates, with respective peak learning rates 10^{-3}, 2×10^{-4}, and 3×10^{-4}. The Flow runs for $80 \times 100 = 8{,}000$ updates at 2×10^{-4}. All stages use AdamW, weight decay 10^{-4}, and cosine decay to 10^{-6}. The training set contains 1,000 numerical pairs and the held-out set contains 64 fields. Each AE update combines a paired batch with an online GRF batch. Flow conditions are drawn approximately equally from paired numerical pressures and dynamic GRFs passed through the frozen forward model.

OCL-PDE, PINO, No-Z, and Conditional EDM+PDE each use the same model for low-fidelity and clean-anchor observations. Model parameters remain frozen during evaluation and pressure refinement. DGenNO is evaluated only in the setting without clean anchors.

After these AE stages, we fine-tune the latent encoder and decoder's latent correction path for 40 epochs, with 100 updates per epoch. The forward branch, observation-based base decoder, and variance head remain frozen. Each update reconstructs the same coefficient field from two independently sampled pressure observations at a shared noise level. Alongside these reconstruction losses, the reference penalty $5 \times 10^{-4}\,\text{mean}[((\mu - \mu^{\text{ref}})/s)^2]$ keeps the adapted latent mean close to the frozen source encoder's mean; s is the source mean's per-channel RMS over batch and spatial dimensions, floored at 0.05. Its weight ramps up over the first five epochs. Because both observations encode the same complete field g, the paired-latent consistency term is identically zero. Validation selects epoch 39. The selected AE is frozen while training the single mixed-condition Flow, and the resulting AE–Flow pair is used for both low-fidelity and clean-anchor evaluations.

Conditional EDM+PDE is trained for 110 epochs, sampling anchor coverage from 0–50% and noise on the remaining observations from 0–3%.

OCL-PDE uses 50 Euler steps in both evaluations, with $(N, K) = (64, 64)$ for low-fidelity observations and $(16, 16)$ for clean-anchor refinement. Field, noise, Flow, and anchor-order seeds are 49091, 59091, 69091, and 820812, respectively. The latter evaluation runs 20 refinement rounds; its full protocol is given in Appendix D.6.

D.3 GRF PRESSURE REFINEMENT WITH CLEAN ANCHORS

Refinement alternates between correcting the native pressure u using PDE and observation errors, and reconstructing z and a from the corrected condition using the frozen Flow and decoder. Only u is updated by gradients; model parameters, z, and a are not gradient-optimized. Algorithm 4 summarizes the procedure for one field. It uses the encoder E_{θ_E}, decoder D_{θ_D}, and Flow endpoint map Φ_{θ_F} from Section 3.3; k indexes refinement rounds and j indexes the K fixed Gaussian sources. The condition y_k uses normalized native pressure and masks for the Flow, and lifted pressure with the decoder validity mask, as described with Eq. (26).

Algorithm 4 GRF pressure refinement with clean anchors for OCL-PDE

Require: Noisy pressure u_{noisy}, clean-anchor values u_{clean} and mask M_H, frozen $(E_{\theta_E}, D_{\theta_D}, v_{\theta_F})$, sample count K, and round count R
Ensure: Refined pressure and coefficient (u, a)
Set $u_0 = (1 - M_H)u_{\text{noisy}} + M_H u_{\text{clean}}$ and $u_0|_{\partial\Omega} = 0$.
Draw fixed sources $z_0^{(j)} \sim \mathcal{N}(0, I)$, $j = 1, \dots, K$.
Form y_0 from u_0 and compute $z_{1,0}^{(j)} = \Phi_{\theta_F}(y_0; z_0^{(j)})$.
Reconstruct the initial coefficient $a_0 = K^{-1} \sum_{j=1}^{K} D_{\theta_D}(z_{1,0}^{(j)}, y_0)$.
Compute $u_{\text{enc},0} = \text{Down}_{32}(E_y(\log a_0, \boldsymbol{x}, f))$; impose zero-boundary and clean-anchor values.
for $k = 0, \dots, R - 1$ **do**
    Start from $u_{\text{enc},0}$ if $k = 0$, otherwise from u_k; enforce the pressure constraints.
    Hold a_k fixed; use the losses in Eq. (27) to compute d_k by Eq. (28).
    Find an accepted u_{k+1} by projected backtracking along d_k.
    if no step is accepted **then**
        return (u_k, a_k).
    end if
    Form the updated condition y_{k+1} from u_{k+1}.
    Update $z_{1,k+1}^{(j)} = \Phi_{\theta_F}(y_{k+1}; z_0^{(j)})$, $j = 1, \dots, K$.
    Update $a_{k+1} = K^{-1} \sum_{j=1}^{K} D_{\theta_D}(z_{1,k+1}^{(j)}, y_{k+1})$.
end for
return (u_R, a_R).

Initialization and reconstruction. The initial coefficient a_0 is reconstructed from the mixed observation u_0 by the Flow and decoder before the encoder is used. Here E_y is the response branch of E_{θ_E}, and Down_{32} restricts its pressure prediction to the native 32×32 observation grid. We use $K = 16$ standard Gaussian sources and 50 Euler steps for each Flow integration, reusing the same sources across all rounds and anchor fractions. Subsequent rounds start from the previously accepted pressure u_k; later encoder predictions are diagnostics only and never replace this iterate.

Pressure losses and update direction. We optimize the current pressure values u_i on the native 32×32 observation grid, while the original noisy measurements $u_{\text{noisy},i}$ remain fixed. Let F denote the interior non-anchor nodes where updates are allowed; clean anchors and the zero boundary are held fixed. With a_k fixed and interpolated bilinearly to this grid, $R_{\text{FV}}(u; a_k, f)$ is the discrete strong-form Darcy residual $-\nabla \cdot (a_k \nabla u) - f$. We evaluate it with a conservative five-point finite-volume stencil: each face coefficient is the arithmetic mean of its two neighboring nodal coefficients, and pressure differences across neighboring nodes determine the face fluxes (Eymard et al., 2000). The PDE loss averages squared residuals after excluding the two outermost grid layers. The observation loss penalizes deviations of the current pressure from the original noisy measurements at the free

nodes:

$$\mathcal{L}_P(u; a_k) = \mathrm{mean}\big(R_{\mathrm{FV}}(u; a_k, f)^2\big)\,, \qquad \mathcal{L}_D(u) = \mathrm{mean}_{i\in F}\left(\frac{u_i - u_{\mathrm{noisy},i}}{s_u}\right)^2. \tag{27}$$

Thus, $\mathcal{L}_D$ is a normalized MSE between the current coarse-grid pressure and the fixed noisy data. It acts as a data-fidelity penalty, discouraging the PDE-driven update from moving too far from the available measurements. Clean-anchor values are imposed as hard constraints rather than included in this loss. The first update starts from the constrained encoder prediction; later updates start from the previously accepted pressure, without replacing it by a new forward prediction. The residual is evaluated using physical pressure u, while the optimization variable is normalized pressure u/s_u. Writing ∇_F for this gradient restricted to free nodes, we normalize the two components separately by their root mean square (RMS), combine them with weights 0.95 and 0.05, and normalize the result once more:

$$\widetilde{d}_k = -\left(0.95\frac{\nabla_F\mathcal{L}_P}{\mathrm{RMS}_F(\nabla_F\mathcal{L}_P)} + 0.05\frac{\nabla_F\mathcal{L}_D}{\mathrm{RMS}_F(\nabla_F\mathcal{L}_D)}\right), \qquad d_k = \frac{\widetilde{d}_k}{\mathrm{RMS}_F(\widetilde{d}_k)}, \tag{28}$$

with safe denominators for zero components. No gradient passes through the coefficient reconstruction or any network during this pressure step.

Constraints, acceptance, and stopping. Before computing the direction and after proposing each step, we enforce zero boundary values, exact clean anchors, and the trust constraint $\mathrm{RMS}_F((u - u_{\mathrm{noisy}})/s_u) \le \rho$. The initial step size in normalized pressure is $\eta_0 = 0.05\rho$; in physical units, a candidate adds $s_u\eta d_k$ to the constrained starting pressure. Backtracking tests this step and at most 20 successive halvings, accepting only a strict decrease in

$$\mathcal{J}_k(u) = 0.95\frac{\mathcal{L}_P(u; a_k)}{\mathcal{L}_{P,\mathrm{pre}}} + 0.05\frac{\mathcal{L}_D(u)}{\max(\rho^2, 10^{-12})}, \tag{29}$$

where $\mathcal{L}_{P,\mathrm{pre}}$ is the PDE loss at the constrained starting pressure, lower-bounded by 10^{-12} and held fixed during the search. If no step is accepted, that field keeps its previous (u_k, a_k) for all remaining rounds; other fields continue independently. After an accepted step, the Flow and decoder recompute z and a as shown in the algorithm. We use $R = 20$ rounds and report rounds 0 and 20. For PINO, the inverse FNO reconstructs $a_k = G_{\mathrm{PINO}}(u_k)$ at each round. Its own forward FNO supplies $F_{\mathrm{PINO}}(a_0)$ to initialize the first pressure update; subsequent updates start from the previously accepted u_k. This test-time pressure correction uses the finite-volume residual in Eq. (27), while PINO training uses the CSRBF weak loss. All network weights remain fixed, and no numerical Darcy solver is called inside the refinement loop.

D.4 DGenNO results using the original optimization procedure

We recover continuous Darcy coefficients from coarse pressure observations without clean anchors using our trained generators and DGenNO's original Beta-MLP parameterization and optimization schedule (Zang & Koutsourelakis, 2025). The test-time optimization uses 1,000 Adam steps without multiple initializations or L-BFGS. This supplements the direct-latent optimization used in the main comparison; the two procedures are specified in Appendix B.3. The original DGenNO study uses a different continuous-coefficient family and does not evaluate our 128×128 GRF Darcy setting. Our models are trained for this setting and provide a reference comparison (Table 7).

Table 7: Continuous Darcy results using the original DGenNO optimization procedure on the coarse pressure grid ($N = 64$ per setting). Errors are relative L_2.

Method	d	ρ	Coefficient a	Solver-u	Time (s)
DGenNO	1024	0	0.71855 ± 0.26673	0.19793 ± 0.08747	24.01364 ± 3.89045
DGenNO	1024	0.01	0.66729 ± 0.25667	0.20109 ± 0.10271	24.22199 ± 5.68484
DGenNO	1024	0.03	0.71316 ± 0.24072	0.20244 ± 0.08668	25.00300 ± 4.82461

D.5 Forward prediction

The encoder also predicts the pressure directly from the permeability, $a \mapsto \hat{u}$. This secondary capability is distinct from solving the Darcy equation using a reconstructed a. Results use the same

64 held-out fields, with both the coefficient input and pressure output evaluated on the full 128×128 grid. The OCL-PDE/No-Z row uses the retrained shared forward branch. This branch has the lowest pressure relative L_2 error among the evaluated forward models.

Method	Direct $a \to u$ relative $L_2 \downarrow$
PINO	0.003822
DGenNO ($d = 1024$)	0.01602
OCL-PDE/No-Z shared branch	0.002689

D.6 Refinement evaluation with clean anchors

Evaluation protocol. OCL-PDE, PINO, No-Z, and Conditional EDM+PDE use their clean-anchor models throughout this study, with the same checkpoint reused across anchor fractions and refinement rounds. The study uses 16 held-out fields at $\rho \in \{1\%, 3\%\}$. The main table uses 450 clean anchors among 900 interior nodes; the complete scan uses nested fractions $0, 5, \ldots, 50\%$. A seeded farthest-point traversal (Gonzalez, 1985) defines a spatially covering anchor order, with each fraction a prefix of the next. Field, noise, Flow, and anchor-order seeds are 49091, 59091, 69091, and 820812, respectively. Clean anchor values are the sole explicit clean-pressure input. The clean coefficient truth, clean non-anchor pressures, and independent numerical-solver pressure errors are used only for evaluation, never for the update.

Complete anchor sweep. We vary clean-anchor coverage from 0% to 50% of the 900 eligible interior pressure locations and compare coefficient error before and after refinement. Before refinement, OCL-PDE is already competitive: coefficient error is lowest at 1% noise and close to No-Z at 3% noise. Refinement starts to reduce OCL-PDE's error at 10% anchor coverage; its final error is lowest from 10% coverage at 1% noise and from 15% coverage at 3% noise. With more anchors, the improvement grows, reaching 34.9% and 54.6% relative error reductions at 50% coverage. At 0–5% coverage, refinement instead increases OCL-PDE's error, showing that sufficient clean information is needed for reliable correction (Tables 8–11).

Table 8: GRF coefficient relative L_2 at $\rho = 1\%$ with 0–20% clean-anchor coverage. Each cell reports round 0 $\to$ round 20; parentheses give the number of clean anchors. Lower is better.

Method	0% (0)	5% (45)	10% (90)	15% (135)	20% (180)
PINO	0.04029 → 0.04558	0.04012 → 0.04403	0.04009 → 0.04267	0.03989 → 0.04064	0.03960 → 0.03944
Conditional EDM+PDE	0.10212 → 0.11001	0.10014 → 0.10678	0.09885 → 0.10398	0.09716 → 0.10090	0.09528 → 0.09776
No-Z ablation	0.04013 → 0.04370	0.03992 → **0.04182**	0.03968 → 0.04034	0.03902 → 0.03698	0.03863 → 0.03530
OCL-PDE	0.03967 → **0.04297**	0.03934 → 0.04217	0.03910 → **0.03884**	0.03850 → **0.03412**	0.03798 → **0.03027**

Table 9: GRF coefficient relative L_2 at $\rho = 1\%$ with 25–50% clean-anchor coverage. Each cell reports round 0 $\to$ round 20; parentheses give the number of clean anchors. Lower is better.

Method	25% (225)	30% (270)	35% (315)	40% (360)	45% (405)	50% (450)
PINO	0.03946 → 0.03874	0.03936 → 0.03784	0.03914 → 0.03729	0.03892 → 0.03705	0.03875 → 0.03686	0.03836 → 0.03669
Conditional EDM+PDE	0.09441 → 0.09654	0.09468 → 0.09653	0.09486 → 0.09633	0.09525 → 0.09651	0.09647 → 0.09754	0.09808 → 0.09912
No-Z ablation	0.03823 → 0.03436	0.03790 → 0.03303	0.03738 → 0.03221	0.03692 → 0.03192	0.03670 → 0.03168	0.03605 → 0.03151
OCL-PDE	0.03756 → **0.02833**	0.03727 → **0.02549**	0.03663 → **0.02360**	0.03581 → **0.02324**	0.03527 → **0.02271**	0.03456 → **0.02249**

Table 10: GRF coefficient relative L_2 at $\rho = 3\%$ with 0–20% clean-anchor coverage. Each cell reports round 0 $\to$ round 20; parentheses give the number of clean anchors. Lower is better.

Method	0% (0)	5% (45)	10% (90)	15% (135)	20% (180)
PINO	0.06365 → 0.06083	0.06292 → **0.05531**	0.06223 → **0.05116**	0.06117 → 0.04530	0.05990 → 0.04240
Conditional EDM+PDE	0.10497 → 0.12520	0.10276 → 0.11878	0.10129 → 0.11278	0.09977 → 0.10632	0.09779 → 0.10037
No-Z ablation	0.05806 → **0.06051**	0.05780 → 0.05700	0.05741 → 0.05361	0.05668 → 0.04975	0.05594 → 0.04705
OCL-PDE	0.06092 → 0.06599	0.06041 → 0.06072	0.05994 → 0.05442	0.05927 → **0.04451**	0.05827 → **0.03895**

Table 11: GRF coefficient relative L_2 at $\rho = 3\%$ with 25–50% clean-anchor coverage. Each cell reports round 0 $\rightarrow$ round 20; parentheses give the number of clean anchors. Lower is better.

Method	25% (225)	30% (270)	35% (315)	40% (360)	45% (405)	50% (450)
PINO	0.05858 → 0.04053	0.05782 → 0.03887	0.05667 → 0.03791	0.05548 → 0.03750	0.05395 → 0.03727	0.05252 → 0.03694
Conditional EDM+PDE	0.09668 → 0.09789	0.09680 → 0.09735	0.09676 → 0.09642	0.09698 → 0.09636	0.09812 → 0.09739	0.09927 → 0.09893
No-Z ablation	0.05516 → 0.04560	0.05487 → 0.04344	0.05411 → 0.04221	0.05316 → 0.04156	0.05204 → 0.04085	0.05127 → 0.04034
OCL-PDE	0.05731 → **0.03597**	0.05691 → **0.03019**	0.05625 → **0.02633**	0.05488 → **0.02566**	0.05317 → **0.02457**	0.05239 → **0.02381**

D.7 Robustness to noisy coarse pressure observations

We assess continuous-coefficient Darcy recovery when all interior coarse-grid pressure observations are subject to additive Gaussian noise, without clean anchors or test-time pressure refinement. On the aligned 32×32 grid, the 900 interior values satisfy $y_i = u_i + \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0, (\rho s_u)^2)$ and s_u is estimated from the training set. We evaluate $\rho \in \{0, 0.01, 0.03\}$, with $\rho = 0$ providing the clean control; the known zero boundary remains exact. Results are averaged over 64 held-out fields. This experiment complements the clean-anchor study in the main text by measuring sensitivity to noise when no higher-quality measurements are available. OCL-PDE, PINO, No-Z, and Conditional EDM+PDE use the same checkpoints as in the clean-anchor evaluation, with zero clean anchors and no pressure refinement. OCL-PDE uses $K = 64$ samples and 50 Euler steps; Conditional EDM+PDE uses its final raw weights, $K = 64$ samples, and 25 Karras–Heun steps. Both average samples in physical coefficient space. Solver-u is evaluated on all 64 fields at each noise level.

Table 12: Robustness of continuous Darcy recovery to additive pressure noise on the aligned 32×32 pressure grid. Independent Gaussian noise with standard deviation ρs_u is added to all interior pressure values; the boundary remains zero. The case $\rho = 0$ is noise-free. Reported coefficient and solver-u errors are means over 64 held-out fields. For each method with recorded timing, test time is the mean per-field inference time over all three noise levels, excluding numerical Darcy solves; "–" denotes unavailable timing. Solver-u denotes an independent numerical Darcy solve from each reconstructed coefficient. Lower is better.

Method	a relative L_2			Solver-u relative L_2			Mean time (s/field)
	$\rho = 0$	$\rho = 0.01$	$\rho = 0.03$	$\rho = 0$	$\rho = 0.01$	$\rho = 0.03$	
PINO	0.03561	0.03986	0.06239	0.00430	0.00471	0.00711	0.00141
DGenNO ($d = 1024$)	0.15056	0.14958	0.15234	0.03358	0.03316	0.03405	99.74
Conditional EDM+PDE	0.09575	0.09636	0.09943	0.02633	0.02637	0.02674	12.25
No-Z ablation	0.02742	0.03858	**0.05704**	0.00318	0.00371	0.00629	0.01571
OCL-PDE	**0.01807**	**0.03686**	0.05740	**0.00192**	**0.00334**	**0.00620**	2.541

Pressure error is lowest for OCL-PDE at all three noise levels, while coefficient error is lowest at $\rho = 0$ and 1% (Table 12). At $\rho = 3\%$, No-Z has a slightly lower coefficient error (0.05704 versus 0.05740). OCL-PDE's coefficient error rises from 0.01807 without noise to 0.05740 at 3% noise, but remains below all external baselines at every tested noise level. Pressure error increases from 0.00192 to 0.00620 while retaining the lowest error. Thus, reconstruction degrades with noise but remains controlled over the tested range without clean anchors or refinement.

E Binary-Darcy implementation and additional results

E.1 Latent construction

The latent branch directly encodes the complete phase geometry. Let $S(a)$ be the signed Euclidean distance to the phase boundary, positive in the high-permeability phase, divided by eight pixels and clipped to $[-1, 1]$ (Curless & Levoy, 1996). We use the full-field input $h = S(a)/2$ and apply zero-input centering to the latent mean:

$$h = \frac{S(a)}{2}, \qquad \mu_z = E^z_{\theta_E}(h) - E^z_{\theta_E}(0). \tag{30}$$

The latent has shape $4 \times 16 \times 16$. The signed-distance transform is used only by the latent encoder; the forward branch takes the physical coefficient, and the decoder predicts phase logits.

E.2 DATA, NETWORKS, AND TRAINING

Observation sampling. During training, the random-point observation rate is sampled over the full 2%–10% range, rather than restricted to the three evaluation rates; 2%, 5%, and 10% are used only for formal evaluation and reporting.

Network inputs and reconstruction. Binary Darcy retains the Darcy FNO/CNN/U-Net-style core. The response operator receives the physical coefficient $a \in \{3, 12\}^{B\times1\times128\times128}$, coordinates, and forcing, and returns the full pressure field. The decoder receives z, raw zero-filled pressure $y_q = M_q \odot u \in \mathbb{R}^{B\times1\times128\times128}$, its exact binary mask M_q, coordinates, and forcing. Observations are random interior points with no interpolation. The fixed test masks at 2%, 5%, and 10% are nested and contain 318, 794, and 1588 points, respectively.

The decoder combines an observation-conditioned base branch with a latent correction to predict binary logits, and is trained with binary cross-entropy with logits. A sigmoid gives the high-phase probability; we average it over $K = 16$ samples and threshold the mean at 0.5.

Flow conditioning. The decoder receives pressure in physical units. The Flow divides its pressure input once by the training-set standard deviation s_u and receives

$$\eta_{\text{bin}} = [y_q/s_u, M_q] \quad \text{on } 128^2, \tag{31}$$

The Flow is conditioned on observed pressure and its mask. Its FPN-inspired condition extractor maps these inputs to the 16×16 latent grid. The velocity backbone uses width 192, four spatially modulated residual blocks, one four-head self-attention block, and four further residual blocks.

Training and evaluation settings. The AE has 30,313,272 parameters; the 20-parameter log-variance head and 10,503,668-parameter Flow give a total of 40,816,960. Training uses a frozen physical forward branch and 1,000 paired training fields. The base is trained for 100 epochs with 100 updates per epoch and batch size 8, using paired numerical pressure observations and a $5\times$ binary cross-entropy loss. With the base and forward branch frozen, the full-z encoder and decoder's latent path are trained for 100 epochs with 100 updates per epoch and batch size 4, using paired and generated fields with reconstruction and PDE losses. The autoencoder is then frozen for 100 epochs of pressure-and-mask-conditioned Flow training, with 100 updates per epoch, batch size 8, four observation views, and equal probabilities of paired numerical and dynamically predicted pressures.

Evaluation uses the same 64 held-out fields for all methods; OCL-PDE generates $K = 16$ samples with 50 Euler steps. PINO uses epoch-110 weights, DiffusionPDE the epoch-350 EMA checkpoint, and DGenNO its final $d = 1024$ weights.

E.3 ADDITIONAL RECONSTRUCTION METRICS

We assess interface recovery using Boundary F1, ASSD, and HD95 on the same 64 held-out fields and three sparse observation masks as in the main experiment. OCL-PDE achieves the highest Boundary F1 and the lowest mean ASSD and HD95 at all three observation rates (Table 13).

Table 13: Additional Binary Darcy metrics at 2%, 5%, and 10% observation rates. Boundary F1 uses a one-pixel tolerance; ASSD and HD95 are measured in pixels. Bold values are best within each observation rate.

Method	2% (318 points)			5% (794 points)			10% (1588 points)		
	Boundary F1 ↑	ASSD (px) ↓	HD95 (px) ↓	Boundary F1 ↑	ASSD (px) ↓	HD95 (px) ↓	Boundary F1 ↑	ASSD (px) ↓	HD95 (px) ↓
PINO	0.47774	3.43339	13.20167	0.60188	2.40210	9.76224	0.65318	2.12762	9.99703
DGenNO ($d = 1024$)	0.49796	5.12891	21.74188	0.50092	5.21081	22.72245	0.49330	5.39692	22.19505
DiffusionPDE	0.3540	7.6068	29.0139	0.4270	6.5002	26.5313	0.4493	6.1458	25.1457
No-Z	0.6279	2.8237	13.3906	0.7514	2.0703	10.8046	0.7894	1.8443	9.4692
OCL-PDE	**0.7509**	**1.9840**	**10.0763**	**0.8721**	**1.2133**	**6.4314**	**0.8943**	**0.9803**	**5.2465**

The median field-wise HD95 values for OCL-PDE are 5.8310, 4.0246, and 3.0000 pixels at 2%, 5%, and 10% observation rates, respectively, compared with 9.21954, 6.32456, and 5.38516 pixels for PINO. Its mean IoU values are 0.93492, 0.95610, and 0.96061, and mean Dice scores are 0.96615, 0.97744, and 0.97982, respectively.

E.4 DGenNO WITH THE ORIGINAL INFERENCE PROCEDURE

We apply DGenNO's original inverse inference procedure to binary coefficient recovery from random pressure observations. The original Darcy experiments use a 29×29 grid (Zang & Koutsourelakis, 2025); we train the adapted model on our 128×128 data to provide a reference comparison (Table 14).

Table 14: DGenNO with the original inference procedure for Binary Darcy from random pressure observations ($N = 64$ per setting). Entries are mean $\pm$ sample standard deviation. Both errors are reported as ratios.

Method	d	Observed	Hard error	Solver-u	Time (s)
DGenNO	1024	2%	0.16463 ± 0.07715	0.13248 ± 0.06821	25.21020 ± 5.67132
DGenNO	1024	5%	0.18134 ± 0.10124	0.14612 ± 0.08035	26.16990 ± 2.63666
DGenNO	1024	10%	0.17399 ± 0.07537	0.15497 ± 0.09822	24.58750 ± 5.65273

E.5 FORWARD PREDICTION

We evaluate the same 64 held-out coefficient fields on the full 128×128 pressure grid. DiffusionPDE uses the epoch-350 EMA joint (a, u) model to predict pressure conditioned on the full ground-truth coefficient field, without pressure observations, using one 2,000-step sample per field. We compare the generated pressure with the ground-truth pressure. OCL-PDE achieves the lowest pressure relative L_2 error among the evaluated methods.

Method	Pressure relative $L_2 \downarrow$
PINO	0.01447
DGenNO ($d = 1024$)	0.05679
DiffusionPDE (conditional joint sampling)	0.154958
No-Z (matched)	0.028866
OCL-PDE	0.012852

E.6 ROBUSTNESS AND DIAGNOSTIC STUDIES

Binary-Darcy direct noise stress test. We compare the clean-trained OCL-PDE, No-Z, and PINO models from the main experiment without retraining or test-time refinement. All methods use the same 64 held-out fields, nested pressure masks at 2%, 5%, and 10% observation density, and identical noisy pressure inputs. OCL-PDE uses $K = 16$ samples and 50 Euler steps. Gaussian noise is added only at observed interior points; unobserved locations remain zero, with no interpolation or noise-level token. For the reported target PSNR levels $P \in \{80, 50\}$ dB, we set $\sigma = R_u 10^{-P/20}$, where the global pressure range $R_u \approx 0.01742$ is estimated from training data only. The corresponding noise standard deviations are 1.741593×10^{-6} and 5.507402×10^{-5}, respectively. Each field uses paired OCL-PDE sampling seeds and a shared standard-normal noise grid across densities and noise levels. The clean No-Z and PINO probability arrays exactly reproduce the saved main-experiment outputs.

Table 15: Binary Darcy reconstruction under observation noise without retraining or refinement. Mean hard pixel errors over 64 fields are reported as fractions.

Observation density	Method	Clean	80 dB	50 dB
2%	PINO	0.068003	0.067999	0.068479
	No-Z	0.046590	0.046590	0.047270
	OCL-PDE	**0.035237**	**0.035226**	**0.036256**
5%	PINO	0.050697	0.050689	0.050801
	No-Z	0.034540	0.034532	0.035040
	OCL-PDE	**0.023565**	**0.023566**	**0.024655**
10%	PINO	0.045720	0.045725	0.045819
	No-Z	0.031425	0.031434	0.031753
	OCL-PDE	**0.021048**	**0.021038**	**0.021707**

Hard pixel error changes little at the reported 80 and 50 dB noise levels. OCL-PDE has lower error than both No-Z and PINO at every observation density under these two noise levels (Table 15). At 50 dB, its absolute error increase is 0.001019, 0.001090, and 0.000659 at 2%, 5%, and 10% density, respectively.

Binary-Darcy DiffusionPDE sanity check. Our DiffusionPDE model is trained on only 1,000 supervised field pairs. Its weaker pressure-only recovery motivates a check of whether the learned joint model can use additional information; this limited-data evaluation differs from the original DiffusionPDE benchmark (Huang et al., 2024). We use the same epoch 350 EMA checkpoint as in the Binary Darcy main experiment to examine the effect of additional coefficient observations. We retain the held-out field and random pressure masks used in the main comparison (Figure 5). At each sparsity, we compare pressure-only recovery with recovery using 500 additional coefficient values at fixed random locations (mask seed 1). Both runs use 2,000 sampling steps and the same initial sampling noise.

Adding coefficient observations reduces both hard pixel error and solver-u error at all three sparsities for this field (Figure 8). The substantial improvement shows that the model responds to additional coefficient information, providing evidence against a collapsed joint prior. This check supports the functioning of the trained model, but does not by itself rule out all training or implementation issues.

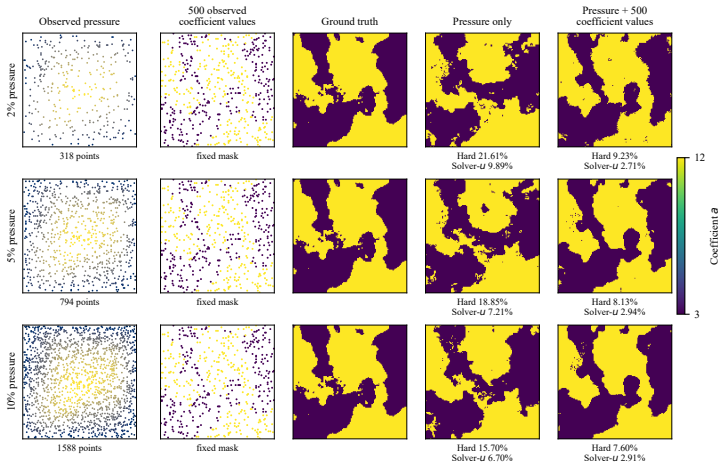


Figure 8: DiffusionPDE coefficient-observation sanity check with the main experiment's epoch 350 EMA checkpoint. Rows show the same held-out field at 2%, 5%, and 10% random pressure observation rates. Columns show observed pressure, the 500 fixed coefficient observations, ground truth, pressure-only recovery, and recovery with both types of observations. Reconstructed coefficients are thresholded at the phase midpoint; annotations report hard pixel error and solver-u relative L_2 error from an independent numerical solve.

F SUPPLEMENTARY ABLATION RESULTS

F.1 COMPLEMENTARY-LATENT INTERVENTIONS

Across the four directed reconstructions from two fixed pairs, the mean matched relative L_2 error is 0.0261. It increases to 0.7233 when U is swapped, 0.7862 when Z is swapped, 0.9968 when Z is set to zero, and 0.5527 when U is set to zero (Figure 9). These two-pair results are mechanism evidence rather than population statistics.

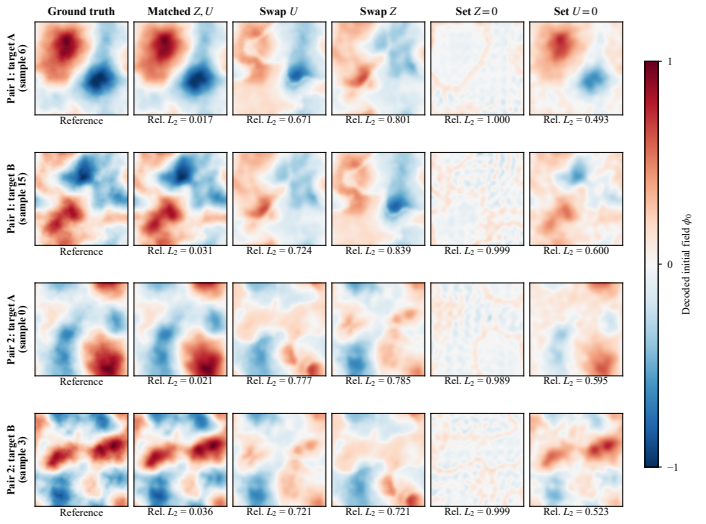


Figure 9: Full Allen–Cahn decoder-branch intervention for two fixed test pairs: trajectories 6 and 15 in the first two rows, and trajectories 0 and 3 in the last two rows. Columns show the reference field, matched reconstruction, exchanged observation, exchanged latent, zero latent, and zero observation. The plot labels U and Z denote observation y and latent z, respectively. Each reconstruction is annotated with its relative L_2 error.

F.2 WHAT THE LATENT ADDS TO A FIXED-OBSERVATION RECONSTRUCTION

Protocol and scope. We perform inference-only diagnostics with one trained checkpoint pair per task (training seed 0), using all 20 held-out Allen–Cahn trajectories and all 64 held-out GRF Darcy fields. Networks are frozen and evaluated with identical observations and decoder conditions across interventions. Allen–Cahn uses the clean full terminal field at $t = 4$ and reuses the saved formal Flow posterior mean (inference seed 9091). GRF uses its mixed-observation AE/Flow checkpoints at $\rho = 0$ with the native 32×32 pressure grid (900 interior measurements), no clean anchors, and no pressure refinement. Its inference seed is 69091. Both generated estimates use $K = 64$ samples and 50 Euler steps. The GRF diagnostic uses $K = 64$, compared with $K = 16$ in the clean-anchor experiment. Test IDs 0 and 1, the first two in canonical stored order, were fixed for visualization before computing errors.

Binary Darcy uses the AE and Flow from the main experiment, evaluated on the same 64 fields at 2%, 5%, and 10% observation rates. Inference uses $K = 16$, 50 Euler steps, and fixed per-field Gaussian seeds $29091 + i$. Its generated probabilities exactly reproduce the formal test outputs. All three reconstructions share the same AE and observations; figures use the first two stored fields, IDs 0 and 1, at 5% observations.

Three reconstructions from the same decoder. Let $f = \phi(\cdot, 0)$ for Allen–Cahn and $f = g = \log a$ for GRF Darcy. For Binary Darcy, $f = \log a$ is computed from the hard coefficient field $a \in \{3, 12\}$. Write $D_{\theta_D}(z, y)$ for the decoder output in this analysis field, including the $\tanh$ postprocessing for GRF and the identity transform for Allen–Cahn. For Binary Darcy, the decoder logits are converted

to probabilities and thresholded at 0.5 before taking $\log a$. We compare

$$f_0 = D_{\theta_D}(z = 0, y), \qquad f_{\mathrm{orc}} = D_{\theta_D}(z^\star, y), \qquad r^\star = f_{\mathrm{true}} - f_0, \qquad \Delta f_m = \tilde{f}_m - f_0. \quad (32)$$

Here $z^\star = \mu$ is the encoder mean obtained from the ground-truth field, and $r^\star$ is the correction needed beyond f_0. The oracle tests the representation and decoder, uses ground truth, and is not an available inverse estimator or a guaranteed error lower bound. The $D_{\theta_D}(z = 0, y)$ output uses the same checkpoint as the other two reconstructions and is distinct from the separately trained, parameter-matched No-Z baseline. For Allen–Cahn, literal $z = 0$ does not exactly disable the latent branch because its decoder lacks the exact zero-correction centering used in GRF; its error should not be interpreted as the amount of information recoverable from the observation alone.

The Flow estimate uses only the observation and observation-derived conditions. Allen–Cahn averages decoded initial fields. GRF averages in physical coefficient space:

$$\widehat{a}_{\mathrm{gen}} = \frac{1}{K} \sum_{k=1}^{K} \exp G(z^{(k)}, y), \qquad \widehat{g}_{\mathrm{gen}} = \log \widehat{a}_{\mathrm{gen}}. \quad (33)$$

Latents are mapped back from standardized Flow coordinates to encoder coordinates before decoding. Neither a mean latent nor a mean log-field is substituted for Eq. (33). GRF uses the same native-grid Flow checkpoint as the main experiments, without a base-field condition. For Binary Darcy, the Flow point estimate averages the $K = 16$ sigmoid probabilities before thresholding at 0.5 and mapping to $a \in \{3, 12\}$. The diagnostic analysis then uses its logarithm.

Metrics. We report relative L_2 error in the physical field ($\phi(\cdot, 0)$ or a), and in the analysis field f. For a detail-sensitive diagnostic, P_{hi} selects coefficients of an orthonormal two-dimensional DCT-II with zero-based indices satisfying $k_x^2 + k_y^2 \geq 16^2$. We compute

$$E_f = \frac{\|\tilde{f} - f_{\mathrm{true}}\|_2}{\|f_{\mathrm{true}}\|_2}, \qquad E_{\mathrm{hi}} = \frac{\|P_{\mathrm{hi}}(\tilde{f} - f_{\mathrm{true}})\|_2}{\|P_{\mathrm{hi}} f_{\mathrm{true}}\|_2}, \qquad Q = 1 - \frac{\sum_i \|\tilde{f}_i - f_{\mathrm{true},i}\|_2^2}{\sum_i \|f_{0,i} - f_{\mathrm{true},i}\|_2^2}. \quad (34)$$

Thus Q pools squared errors before taking the ratio, and is not clipped. Metrics are accumulated in float64 with a common denominator floor of 10^{-12}; no test field activates this floor. Standard deviations are sample standard deviations across held-out fields, not across training seeds. High-band error measures recovery of the correct spatial details, rather than merely the presence of high-frequency energy.

Results. The latent supplies different corrections across tasks. In GRF Darcy, Flow approximately halves coefficient error ($3.66\% \to 1.81\%$) and removes 75.5% of the pooled log-coefficient squared error ($Q = 0.7553$). The accompanying decrease in high-frequency error ($97.98\% \to 83.80\%$) shows that the latent correction also improves the spatial details measured by this spectral diagnostic (Table 16).

For Binary Darcy, Flow corrects phase assignments: at 5% observations, hard pixel error falls from 4.48% to 2.36%, approximately halving the misclassified area. Across the three observation densities, it removes 36.4–51.5% of the pooled log-coefficient squared error. For Allen–Cahn, Flow reduces initial-field error to 5.90% and removes 99.64% of the zero-latent squared error. Here the latent contributes to the broad initial-field structure as well as finer corrections; the large gain is relative to a zero-latent path that was not separately trained for reconstruction.

The oracle has lower field and high-frequency errors in every setting. For example, GRF high-frequency error falls to 32.11% with the oracle, compared with 83.80% for Flow. Thus the decoder can represent finer reconstructions than those recovered through observation-conditioned sampling. The spatial corrections also recover part of the structure missing from the zero-latent outputs in the two illustrated fields per task (Figures 10–12).

Table 16: Fixed-observation latent diagnostics. Errors are percentages (mean ± sample standard deviation); Q is the pooled analysis-field squared-error reduction relative to $z = 0$. The oracle uses the ground-truth encoder mean μ. Flow uses $K = 64$ samples for Allen–Cahn and GRF Darcy, and $K = 16$ for Binary Darcy. Binary outputs are thresholded after averaging phase probabilities; hard pixel error applies only to Binary Darcy.

Reconstruction	Hard error	E_{physical}	E_f	E_{hi}	Q
Allen–Cahn: $n = 20$, $f = \phi(\cdot, 0)$					
$D_{\theta_D}(z = 0, y)$	–	100.67 ± 0.62	100.67 ± 0.62	188.76 ± 23.43	0
$D_{\theta_D}(z^\star, y)$	–	2.66 ± 0.72	2.66 ± 0.72	55.62 ± 3.47	0.9993
$D_{\theta_D}(z_{flow}, y)$	–	5.90 ± 1.82	5.90 ± 1.82	78.95 ± 3.99	0.9964
GRF Darcy: $n = 64$, $f = \log a$					
$D_{\theta_D}(z = 0, y)$	–	3.66 ± 0.93	9.08 ± 2.51	97.98 ± 2.42	0
$D_{\theta_D}(z^\star, y)$	–	0.57 ± 0.16	1.40 ± 0.45	32.11 ± 2.19	0.9762
$D_{\theta_D}(z_{flow}, y)$	–	1.81 ± 0.50	4.48 ± 1.28	83.80 ± 5.10	0.7553
Binary Darcy: $n = 64$, $f = \log a$, 2% *observations*					
$D_{\theta_D}(z = 0, y)$	5.54 ± 2.56	23.23 ± 5.33	16.33 ± 3.77	121.94 ± 6.16	0.0000
$D_{\theta_D}(z^\star, y)$	0.78 ± 0.54	8.26 ± 3.22	5.81 ± 2.28	49.28 ± 12.62	0.8600
$D_{\theta_D}(z_{flow}, y)$	3.52 ± 1.66	18.50 ± 4.29	13.01 ± 3.05	110.39 ± 5.63	0.3637
Binary Darcy: $n = 64$, $f = \log a$, 5% *observations*					
$D_{\theta_D}(z = 0, y)$	4.48 ± 2.05	20.87 ± 4.82	14.68 ± 3.41	117.02 ± 6.18	0.0000
$D_{\theta_D}(z^\star, y)$	0.66 ± 0.54	7.46 ± 3.32	5.26 ± 2.36	46.05 ± 13.71	0.8519
$D_{\theta_D}(z_{flow}, y)$	2.36 ± 1.19	15.03 ± 3.82	10.57 ± 2.71	96.43 ± 6.59	0.4739
Binary Darcy: $n = 64$, $f = \log a$, 10% *observations*					
$D_{\theta_D}(z = 0, y)$	4.34 ± 2.00	20.50 ± 4.76	14.42 ± 3.39	116.27 ± 5.99	0.0000
$D_{\theta_D}(z^\star, y)$	0.65 ± 0.48	7.51 ± 3.10	5.29 ± 2.20	47.09 ± 13.41	0.8491
$D_{\theta_D}(z_{flow}, y)$	2.10 ± 1.07	14.22 ± 3.59	10.00 ± 2.55	92.54 ± 6.70	0.5148

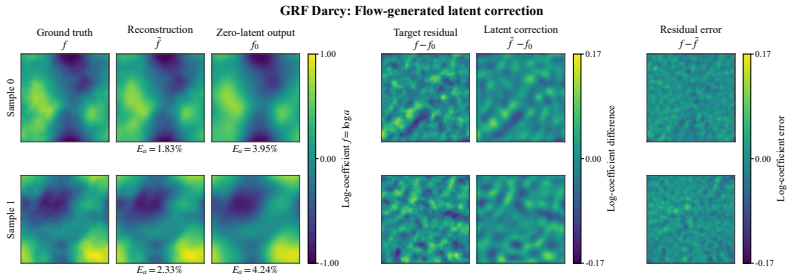


Figure 10: **GRF Darcy latent corrections.** Flow-generated reconstruction and zero-latent base, with the target residual, latent correction, and remaining error in $f = \log a$. We report coefficient reconstruction errors in relative L_2.

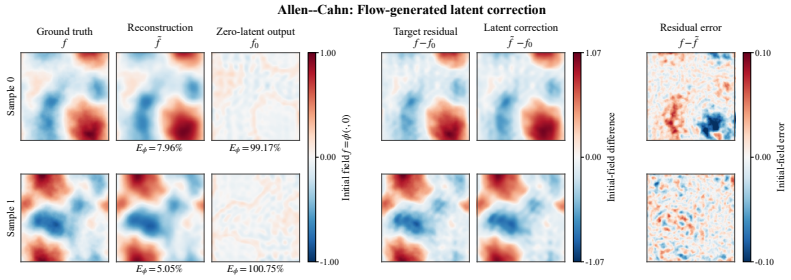


Figure 11: **Allen–Cahn latent corrections.** Flow-generated reconstruction and zero-latent base, with the target residual, latent correction, and remaining error in $f = \phi(\cdot, 0)$. We report initial-field reconstruction errors in relative L_2.

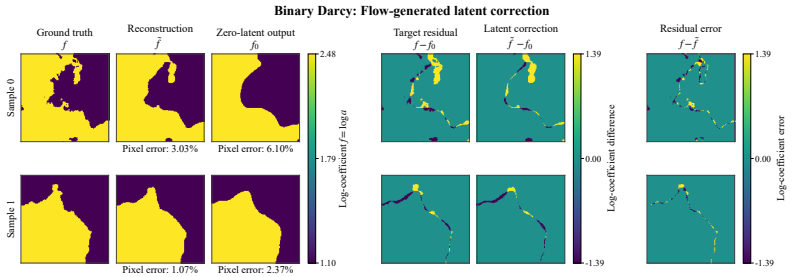


Figure 12: **Binary Darcy latent corrections at 5% observations.** Flow-generated reconstruction and zero-latent base, with the target residual, latent correction, and remaining error in $f = \log a$. We report hard pixel errors.