BEYOND SPATIAL PREDICTION: NONLINEAR OUTPUT REPRESENTATIONS FOR INVERSE SCATTERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Inverse scattering problems represent a severely ill-posed regime where the spatially varying contrast of a medium is recovered from measurements of the scattered wavefield. Typically, machine-learning approaches output reconstructions in a discretized spatial parametrization, where the neural networks predict the contrast at each grid point. In this paper, we study this output choice and build a modular head which reconstructs the contrast field through a learned, lower-dimensional nonlinear parametrization, replacing spatial parametrizations. The network therefore predicts latent coordinates, whose space carries our prior on the structure of our contrast fields. Across three target distributions, three noise levels, and multiple seeds, pretrained latent decoding reduces normalized mean squared error by an average of 34% relative to the spatial-output variants of five leading neural-operator architectures. The gain holds in fairly matched studies, including controls on parameter counts, MACs, and training recipes. In one setting, we observe that a spatial network with $7\times$ the trainable parameters needs a longer training schedule just to reach parity with a nonlinear latent head. We also study the impact of nonlinearity, finding the largest gain over linear latents for sharp contrast fields and less for smooth low-rank fields. Our results show that nonlinear latents serve as a strong representation space for learned inverse scattering methods.

1 INTRODUCTION

Modern research on machine-learning for inverse PDE solving mainly focuses on network architectures, with significant breakthroughs coming from architectures which encode some aspect of the underlying physics, such as Fourier Neural Operators (FNOs) (Li et al., 2021). The backbone is only part of an inverse solver, though; a network's output parametrization decides how the contrast field is built from the learned features. Some inverse scattering architectures focus on output representations as central design elements, but most predict the contrast field directly in spatial coordinates. In this work, we ask the 'latent' question: should inverse scattering networks predict the field in spatial coordinates, a linear basis, or a nonlinear latent space?

The choice of representation is not 'one-size-fits-all' and depends on the target family. Contrast fields with sharp edges and translated features can require many linear modes, motivating nonlinear approximation spaces (Ohlberger & Rave, 2016; Peherstorfer, 2022; Greif & Urban, 2019), while smooth low-rank fields may already be represented effectively in a linear basis. We focus our investigation on identifying if and when nonlinearity pays off over a linear basis or spatial parametrization.

The performance of latent-predicting networks has been researched in microwave, acoustic, and seismic imaging (Shao & Du, 2020; Ahmed et al., 2023; Feng et al., 2024), as well as in the field of operator learning (Seidman et al., 2022; Lanthaler et al., 2023). Our contributions are more general, and our idea is simple: build a latent approximation of the target contrast fields and then train a network to predict the latent coordinates in a modular way, connecting to the output token-grid or feature-map of inverse scattering networks. This allows direct comparison of output heads across backbones. With our method, we capture a prior on the distribution of our contrast fields directly in the output head. This recipe works regardless of whether the nonlinear decoder is pretrained on clean targets or trained jointly with the network. A linear decoder is just a PCA basis. Across five backbones, three synthetic classes of contrast fields, and three noise levels, the nonlinear decoder lowers normalized mean squared error (NMSE) by about a third, winning 44 of 45 conditions

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.

(Section 5.1). Similar results hold for medical-image targets from OrganAMNIST (Yang et al., 2023) (Section 5.2).

The gain does not come from pretraining or extra compute. A nonlinear latent head trained on the same parameter and compute budget as a spatial head wins in every condition for Transolver (Section 6.1). This advantage extends to every decoder training regime tested. Additionally, a nonlinear latent head that predicts as many coordinates as there are pixels (4,096) still outperforms the spatial head, so the gain does not come from compression either (Section 6.2).

When compared to PCA, the gain depends more on the contrast field distribution. Nonlinear coordinates outperform a linear basis on sharp regimes (Shapes, MNIST) but add little performance gain on smooth target distributions that can be well-represented by a linear basis (Section 6.3). We note that the training method may also influence nonlinear performance, as the nonlinear decoder only beats PCA when coordinates are supervised during training on OrganAMNIST (Section 5.2).

We claim the following main contributions:

1. A modular output interface for inverse networks, and a systematic comparison of spatial, linear, and nonlinear outputs across five attention and Fourier backbones (Sections 4–5.2).
2. Controls showing that the gain holds at matched head budgets, without pretraining or code supervision, and without strictly undercomplete coordinates (Sections 6.1–6.2).
3. A characterization of when nonlinearity beats a linear basis, which depends on the structure of the targets and on code supervision (Section 6.3).
4. Error accounting that separates representing the targets from predicting them, and explicit data, capacity, and schedule tradeoffs (Sections 6.4–6.5).

2 Problem setup

Inverse scattering. We consider the inhomogeneous-medium inverse scattering problem at a fixed wavenumber k (Khoo & Ying, 2019; Jiang et al., 2024). An unknown contrast field $x\colon [-1,1)^2 \to \mathbb{R}$ is hit by J incident plane waves $u_j^{\mathrm{in}}(r) = e^{ik\, d_j \cdot r}$ with directions $d_j \in \mathbb{S}^1$. For each j, the total field $u_j = u_j^{\mathrm{in}} + u_j^s$ solves the Helmholtz equation

$$\Delta u_j + k^2\big(1 + x\big)\, u_j = 0 \tag{1}$$

where the scattered field $u_j^s\colon \mathbb{R}^2 \to \mathbb{C}$ satisfies the Sommerfeld radiation condition. Sampling each scattered field at M receivers $r_1, \ldots, r_M$ equally spaced on an encompassing circle of radius 2 defines the forward operator

$$f\colon x \mapsto \big[\, u_j^s(r_m) \,\big]_{m,j} \in \mathbb{C}^{M \times J}, \tag{2}$$

which is nonlinear in x because u_j itself depends on x (multiple scattering). Since the receivers are at a finite distance, f returns the scattered field itself rather than its far-field pattern. The inverse problem is to recover x from noisy measurements $y = f(x) + \varepsilon$.

This problem is ill-posed (Colton & Kress, 2019): with finitely many noisy measurements, infinitely many contrast fields are consistent with a given y up to the noise level, and the reconstruction is highly sensitive to perturbations in y. This makes recovering f^{-1} intractable; instead, we learn a map G_θ from paired samples (x, y) from our target distribution. Our trained neural network G_θ therefore acts as a data-driven, distribution-informed regularized inverse returning an estimate $\hat{x} = G_\theta(y)$. In practice x is represented on a 64×64 grid, so $\hat{x} \in \mathbb{R}^{64\times 64}$.

Output parametrizations. Figure 1 shows three output parametrizations. A backbone B_θ maps measurements y to features, and the output head maps these features to the contrast field. We compare heads within each backbone family listed in Section 4. OFormer's feature readout differs between heads, and the high-data resource controls adjust backbone width (Appendix A.2; Section 6.5). A spatial head H_s predicts the field directly on the grid, a linear head R_a predicts p coordinates in a fixed PCA basis with mean μ and orthonormal columns $\Phi \in \mathbb{R}^{4096\times p}$, and a nonlinear head R_l predicts p coordinates that a decoder D_ϕ maps to the field:

$$\hat{x}_s = H_s(B_{\theta_s}(y)), \qquad \hat{x}_a = \mu + \Phi\, R_a(B_{\theta_a}(y)), \qquad \hat{x}_l = D_\phi(R_l(B_{\theta_l}(y))), \tag{3}$$

with $R_a(\cdot), R_l(\cdot) \in \mathbb{R}^p$. Each predictor is trained separately.

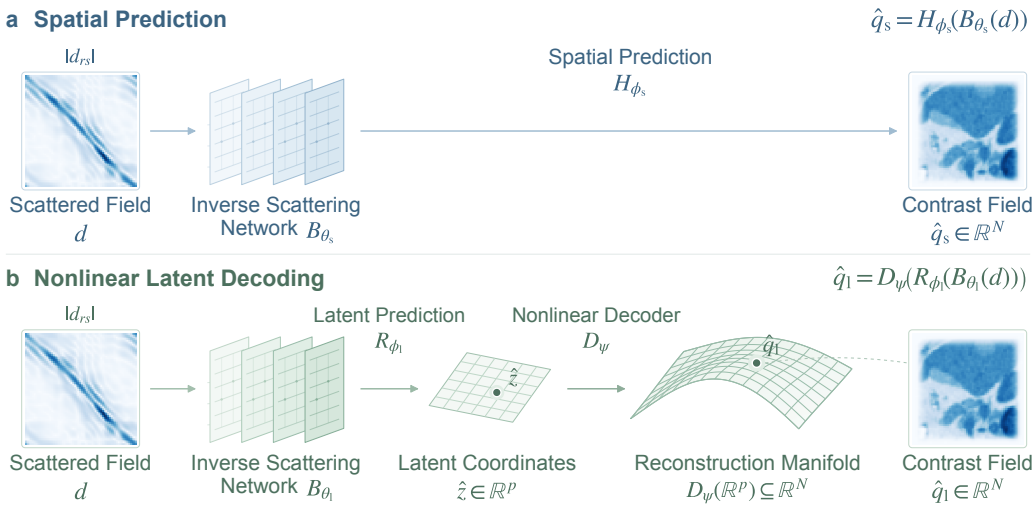

Figure 1: Our modular architecture, adapting a spatial network to latent (linear or nonlinear) coordinates. Each network is trained independently and output the 64×64 contrast field. For nonlinear methods the encoder is used only in training, and the decoder may be pretrained or trained jointly.

3 Learned-coordinate prediction

Coordinate head. Token features are reshaped to the token grid indexed by receiver and incident direction. Feature maps from FFNO enter directly. A 1×1 convolution reduces the channels to 14, adaptive average pooling to an 8×8 grid, and a two-layer MLP of width 256 predicts coordinates. We retain coarse spatial layout before predicting our latent coordinates. Unlike the spatial head, which decodes the token grid locally with convolutions, every latent coordinate head collapses the grid to a global vector from which the field is synthesized; this factorization is the object of our study. For OFormer, the coordinate head takes the normalized trunk features before OFormer's learned-query decoder (Appendix A.2). A convolutional variant replaces the projection and average pooling with weight-shared 3×3 convolutions which downsample the grid to 8×8, followed by a 1×1 convolution and a small MLP. It is used in the matched controls (Section 6.1) and in the high-data benchmark (Section 6.5); Appendix A gives both configurations.

Pretrained recipe. For the nonlinear head of our main benchmark (Section 5.1, Table 1), we first train a VAE on clean targets alone (image MSE plus a KL term weighted by 10^{-4}, 100 epochs) and then freeze its encoder E and decoder D_ϕ, with $p = 256$. The inverse network is trained with

$$\mathcal{L} = \mathrm{MSE}\big(\widetilde{D}(z), \widetilde{x}\big) + \tfrac{1}{2}\,\mathrm{MSE}\big(z, \mu_E(x)\big), \qquad z = R_l(B_{\theta_l}(y)), \tag{4}$$

where $\widetilde{x} = (x - \bar{x})/s_x$ and $\widetilde{D}(z) = (D_\phi(z) - \bar{x})/s_x$ are normalized with the training-target mean $\bar{x}$ and standard deviation s_x, and $\mu_E(x)$ is the posterior mean of the frozen encoder. Each MSE averages over its own coordinates, and gradients flow through the frozen decoder into the coordinate head and backbone. The encoder supplies the code target during training, and is discarded at inference. The spatial head is trained on NMSE, as is the linear decoder PCA_{128} in Table 1. The matched-rank comparisons (Section 6.3) use PCA_{256} and nonlinear decoders with 256 coordinates.

Decoder regimes. We test a frozen pretrained decoder, one pretrained decoder fine-tuned with the inverse network, and a randomly initialized decoder trained jointly with it. As a stricter control in our ablations, the scratch synthesis head attaches a randomly initialized residual decoder to the convolutional coordinate head and trains both jointly with image loss alone. Its parameter count is within 0.02% and its MACs within 0.07% of the spatial head's. Both heads share trunk initializations and data streams, train for 3,000 updates, and use development-selected learning rates (Appendix D). This control removes pretraining, freezing, and code supervision. It does not remove the architectural and optimization differences between a coordinate head and a spatial head. A second control places the frozen pretrained decoder behind the same coordinate head, which uses 44% of the spatial model's trainable parameters and 36% of its inference MACs.

Costs. Inverse networks have between 1.3M and 2.0M parameters. Our pretrained experiments add a frozen 2.28M-parameter decoder and a pretraining stage on clean targets, both of which we count in Section 6.5. Table 24 lists trainable parameters, inference weights, and MACs for every model. Across inverse network backbones we keep the coordinate-head design and, for the pretrained recipe, the target decoder. Appendix A gives code and pseudocode for every component.

4 Experimental setup

Target classes. Contrast fields come from three synthetic target classes: Shapes, MNIST digits (LeCun et al., 1998), and low-rank radial-basis-function fields (LR-RBF). Medical-image-derived OrganAMNIST templates from the official split (Yang et al., 2023) provide a fourth target class. All targets are low-pass 64×64 contrasts, and scattered fields are simulated. Throughout, we fix wavenumber $k = 12$ and use $J = M = 64$ incident plane waves and receivers. Synthetic measurements have additive Gaussian noise at levels $\sigma \in \{0, 0.3, 0.6\}$ relative to the data scale (Appendix A). OrganAMNIST measurements are noise-free.

Backbones. Each output head is attached to five recent operator-learning backbones: four attention-based models, Transolver, Galerkin Transformer, OFormer, and GNOT, and one Fourier model, FFNO (Wu et al., 2024; Cao, 2021; Li et al., 2023; Hao et al., 2023; Tran et al., 2023). We re-implement a compact version based on their respective papers, modified for our data (Appendix A.2). Output-head settings and training protocols are in Appendix A.

Test sets. Headline numbers come from a shared, held-out test set of 2,000 targets per class. Its measurements are simulated from each raw scatterer on a 256×256 grid, and reconstructions are scored against the low-pass target. OrganAMNIST uses its official test split (Appendix A.1).

Metric and uncertainty. We report NMSE, $R = \mathbb{E}\,\|x - \hat{x}\|^2/(64^2\, v)$, where the expectation is over test objects and v is the pixel variance of the scored targets, shared by all heads. For each backbone–class–noise condition c, the relative reduction of the nonlinear head over the spatial head is $g_c = 100\,(1 - \overline{R}_{l,c}/\overline{R}_{s,c})$, where $\overline{R}$ averages NMSE over seeds. The headline number is the mean of g_c over the 45 main-benchmark conditions.

Main-benchmark experiments use three seeds, and matched controls use ten paired seeds. We give a statistical account in Appendices A.4 and B.2 through sign-flip tests, bootstrap intervals, and Holm-corrected target-level tests. Seed SDs measure retraining variability.

5 Results across backbones

5.1 Main benchmark

Nonlinear synthesis improves every backbone. Across our 45 synthetic backbone–class–noise conditions, the pretrained, frozen nonlinear head lowers NMSE by 34.4% on average, winning 44 conditions and 132 of 135 paired-seed comparisons (Table 1, Figure 5a; Appendix Figure 7). The single reversal is GNOT on LR-RBF at $\sigma = 0.6$. On Shapes, the output head matters more than the backbone. Averaged over noise, every backbone with the nonlinear head is more accurate than every backbone with the spatial head.

Where the gain comes from depends on the targets. On smooth fields, predicting any low-dimensional coordinates helps. PCA coordinates beat spatial prediction in 13 of 15 conditions on LR-RBF, and nonlinear coordinates add a further 7.5% (Figure 5b). On Shapes and MNIST, PCA mostly loses to spatial prediction, and the nonlinear head improves on it by 59.4% and 33.1%. This pattern is consistent with the approximation-width motivation of Section 1. Translated, sharp-edged targets need many linear modes, and smooth low-rank fields do not. Section 6.3 repeats the comparison at matched rank with image loss alone.

Noise erodes the gain on some classes but not others. On Shapes, the reduction holds at 48% at every noise level. On MNIST it falls from 24% to 15% as we increase noise, and on LR-RBF from 44% to 30% (Figure 3a–d).

Table 1: **Output representations across five backbones and four target regimes.** NMSE in units of 10^{-2}; bold marks the lowest mean in each solver row and target. The last rows give the VAE's mean NMSE reduction against each baseline and its paired seed wins over spatial. Full errors and SDs: Tables 7 and 8.

	Shapes			MNIST			LR-RBF			OrganAMNIST		
Solver	Spatial	PCA	VAE	Spatial	PCA	VAE	Spatial	PCA	VAE	Spatial	PCA	VAE
Transolver	8.72	13.98	**5.48**	4.11	4.50	**3.13**	7.78	3.79	**3.63**	8.73	6.73	**5.86**
Galerkin	12.12	15.49	**6.47**	4.10	5.70	**3.72**	6.39	5.85	**5.18**	13.80	9.38	**7.85**
OFormer	15.73	14.40	**6.06**	4.89	5.37	**3.57**	10.43	4.75	**4.40**	18.62	7.58	**6.39**
GNOT	11.05	15.15	**6.37**	4.10	5.58	**3.65**	4.96	5.19	**4.58**	12.89	8.42	**7.25**
FFNO	10.02	12.01	**4.62**	3.51	3.92	**2.72**	3.26	1.95	**1.93**	6.91	4.40	**4.09**
Gain vs. spatial (%)		47.8			19.2			36.4			45.3	
Gain vs. PCA (%)		59.4			33.1			7.5			13.2	
Wins vs. spatial		45/45			45/45			42/45			25/25	

Synthetic classes: noise levels averaged within seed, three seeds; gains average per-condition ratios; wins count noise–seed pairs. OrganAMNIST uses five seeds. PCA uses 128 coordinates (256 on OrganAMNIST), the VAE 256. The VAE adds code loss, and so does PCA on OrganAMNIST. Noise averaging hides the GNOT/LR-RBF reversal at noise 0.6.

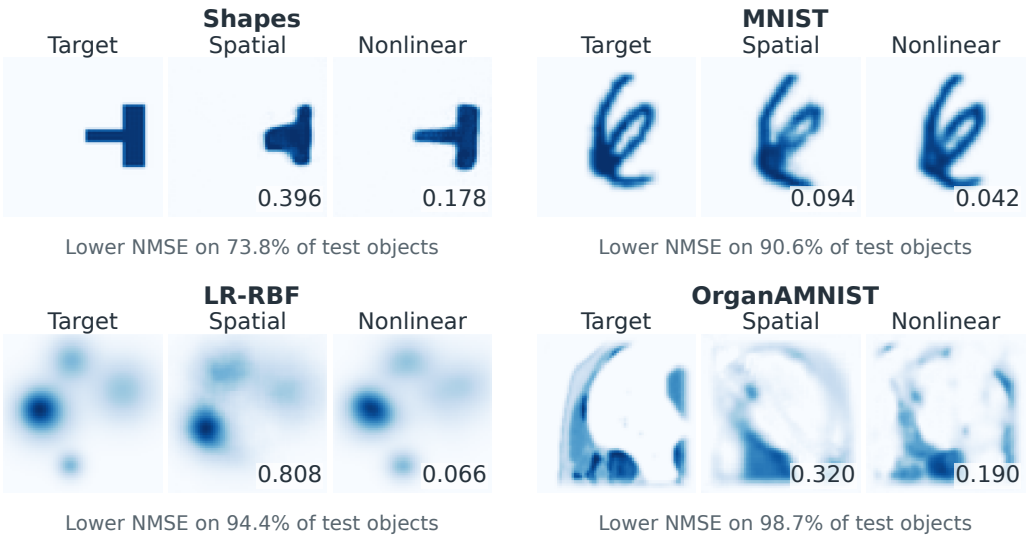

Figure 2: Selected samples from the same test pair, per-object NMSE labeled. The synthetic classes use Transolver seed 42; OrganAMNIST uses its official test split and seed 1001. Samples were picked to be illustrative, aiming to maximize the spatial-minus-latent gap except the hand-selected MNIST digit (rank 17). Table 9 contains more detailed values.

Qualitative observations from nonlinear reconstructions. In the examples of Figure 2, the spatial head's errors are blurred or merged structures. The nonlinear head's reconstructions stay closer to the shapes, strokes, and peaks of the target class. The examples were selected for large gaps. Averaging each object's losses over seeds, nonlinear prediction wins on 74–99% of objects (Figure 3e–h; Table 9).

5.2 Official OrganAMNIST split

OrganAMNIST provides a boundary case for nonlinear decoding. Both coordinate heads improve on spatial prediction, and PCA captures much of the gain. On the official split, averaged over the five backbones, the nonlinear head lowers NMSE by 45.3% relative to spatial prediction and by 13.2% relative to PCA with the same code supervision, and it wins all 25 paired-seed comparisons against each (Table 1; Table 8).

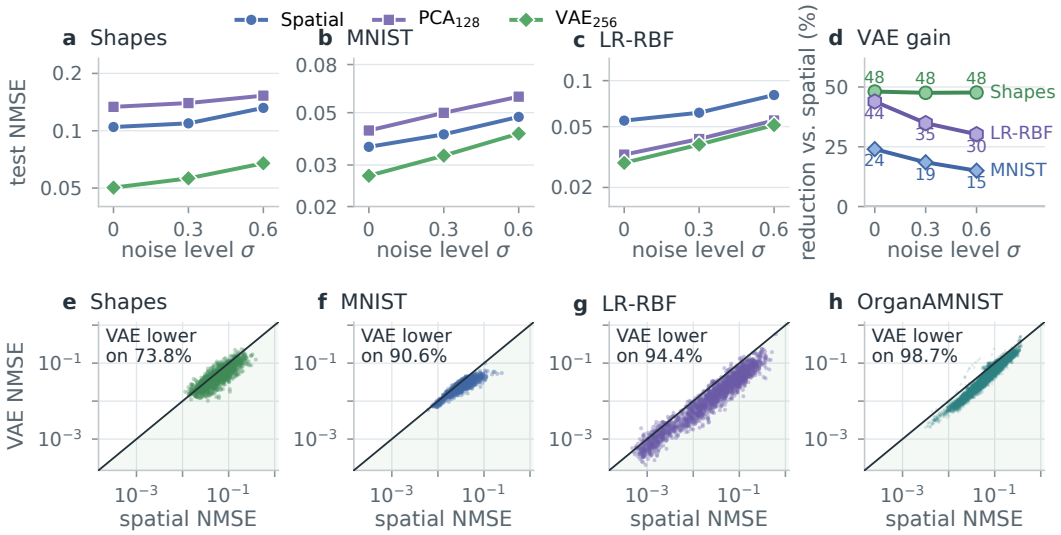

Figure 3: **The gain across noise levels and test objects.** (a–c) Main-test-set NMSE against noise level, averaged over all backbones and seeds (Appendix Figure 8). (d) The VAE head's NMSE reduction relative to the spatial head. (e–h) Per-object NMSE for Transolver on clean measurements, averaged over seeds; points below the diagonal favor the VAE head. Raw data in Table 7.

Table 2: **Scratch nonlinear synthesis at matched head budgets.** Transolver trained with image loss alone and no pretraining. Clean-test NMSE means, gains over the spatial head, and wins over ten paired seeds.

Target	Pairs	Spatial	Scratch	Gain (%)	Wins
Shapes	200	0.4418	0.2375	46.2	10/10
Shapes	3,200	0.0352	0.0127	64.1	10/10
MNIST	200	0.5267	0.3654	30.6	10/10
MNIST	3,200	0.0764	0.0536	29.8	10/10
Smooth fields	200	0.8780	0.7312	16.7	10/10
Smooth fields	3,200	0.2032	0.1223	39.8	10/10
OrganAMNIST	3,200	0.1360	0.0974	28.4	10/10
OrganAMNIST	34,561	0.0729	0.0471	35.3	9/10

The last row uses the full official OrganAMNIST split. Protocols and uncertainty are in Appendix D.

The advantage over PCA depends on supervision. In a ten-seed Transolver control with image loss alone, PCA and the VAE reach nearly the same error (0.0638 and 0.0646). With code supervision, the VAE improves on image-only PCA by 9.6%, and on PCA with the same code loss by 14.3% (Table 8). Here the nonlinear range helps only when its coordinates are supervised. Section 6.4 examines this dependence on coordinate supervision.

5.3 Robustness

The results do not hinge on the test protocol or particular backbones. Scoring the same checkpoints on the low-pass test set gives the same win counts and a 34.0% mean reduction. Refining the solver grid and switching to a boundary-integral solver preserve the ordering in the comparisons we ran (Appendix C.2). IAE-Net, a sixth architecture, shows the same pattern at three model sizes, with the nonlinear head winning all 27 seed comparisons at each (Appendix B.3).

6 Controlled Studies

Here we study what drives the performance of nonlinear representations. We focus on Transolver, and on Shapes for the single-class studies, to isolate the effect of network and target regime. Table 3 summarizes our findings.

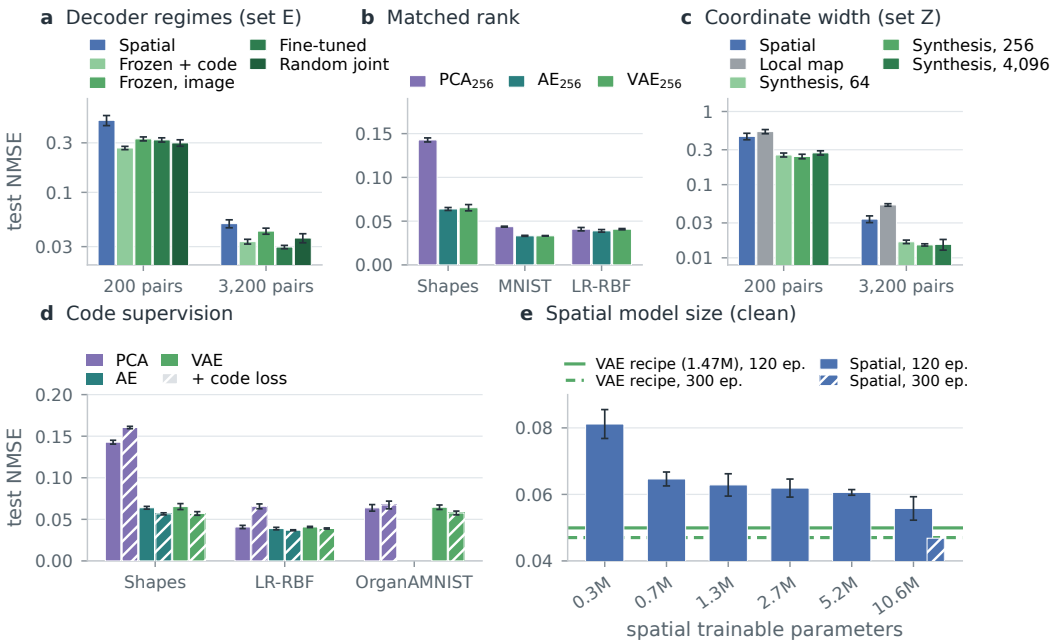

Figure 4: **Controlled studies on the Transolver backbone.** Means and sample SDs over three seeds, or ten in (c) and for OrganAMNIST in (d). Single-class panels use Shapes. (b, d) Main test set, 256 coordinates, noise 0 and 0.6 averaged; (b) uses image loss only. (e) Low-pass test set.

6.1 Matched budgets

The gain is not bought with a larger head or with pretraining. This ablation demonstrates that our gain is not from any mismatch, as we still observe a 17–64% reduction in NMSE in every setting in Table 2, and wins in nearly every matched comparison. If we instead use a frozen pretrained decoder, we see a 5% increase in parameter count but only 44% of the trainable parameters, while using only 36% of the MACs at test time, and the nonlinear head still wins by 34–62% everywhere. Appendix D controls for the choice of hyperparameters.

6.2 Decoder training and coordinate width

The gain does not depend on how the decoder is trained. A frozen pretrained decoder, a fine-tuned one, and a randomly initialized one trained jointly all beat the spatial head (Figure 4a; Appendix D.3). Fine-tuning a pretrained decoder yields the lowest error, 41% below the spatial head with 3,200 pairs. We also see that the dimensionality of the parametrization has little impact, and every size retains at least 85% of the width-256 head's gain over the spatial head (Figure 4c; Appendix D.2). This figure also shows that compression is not what earns the gain. The connection does matter, though, as a budget-matched decoder that uses an 8×8 feature map instead of a global coordinate vector is worse than a spatial head, although channel widths and global mixing also differ.

6.3 Linear versus nonlinear coordinates at matched rank

At matched rank, nonlinear coordinates help where a linear basis struggles. With 256 coordinates and image loss alone, an autoencoder improves over PCA across all synthetic datasets, while a VAE improves over PCA only on Shapes and MNIST and ties it on LR-RBF (Figure 4b). Hence, the variational objective is not what matters. We see that nonlinear coordinates give the biggest benefit on sharp regimes, but marginal gains on smooth fields that are already well-represented by a linear basis. On OrganAMNIST, we notice an advantage only with code supervision (Section 5.2).

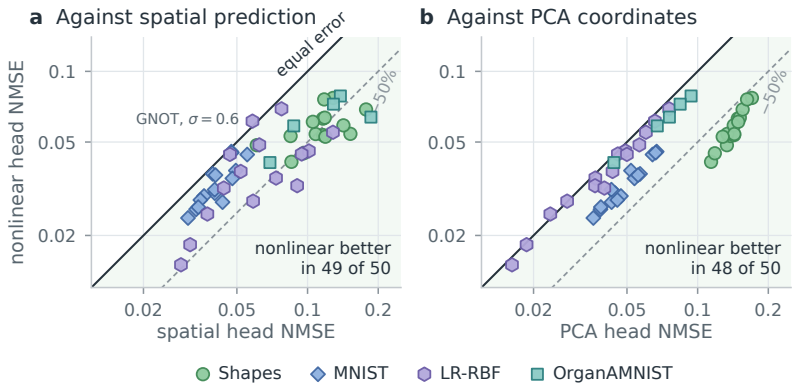

Figure 5: **The gain across the benchmark.** Each point is one backbone, target class, and noise level (seed means), including five OrganAMNIST conditions. Points below the diagonal favor the nonlinear head, in 44 of 45 synthetic conditions and 49 of 50 overall.
PCA as in Table 1.

6.4 Representation floor and code supervision

Better representation explains only part of the gain. Proposition 1 (Appendix E) bounds the representation advantage using only the error of projecting test targets onto the PCA range. At least 53–56% of the gain over PCA remains after subtracting it (Table 18). On OrganAMNIST, the VAE reconstructs the targets less accurately than PCA (0.022 against 0.015), yet predicts better with code supervision. In an exploratory check, projecting a spatial head's predictions onto the frozen decoder's range barely changes their error (Appendix E.5). The coordinates must be learned through the representation, and supervision matters. Code supervision improves the nonlinear heads but degrades PCA (Figure 4d). Appendix E relates reconstruction error to prediction error, and derives how code supervision reweights PCA errors and changes decoder gradients.

6.5 Resource tradeoffs

Our results persist across limited and abundant training data. The nonlinear head wins across backbones regardless of the amount of training data available (Appendix Table 22)—the downside being that clean targets are required to train the decoder. Clean targets can make up for fewer paired samples, with a latent network needing only half as many pairs as a spatial network to reach the same accuracy (Appendix Figure 16a). That advantage disappears when the number of clean targets decreases. A larger spatial model ($7.2\times$ trainable parameters, $2.8\times$ inference weights) trails a latent network at 120 epochs, only reaching comparable error at 300 epochs (Figure 4e; Table 21).

7 Related work

Where the prior lives. Since inverse scattering is ill-posed, learned methods must encode a prior. Classical methods place it in explicit penalties and iterative solvers (Colton & Kress, 2019; Engl et al., 1996). Generative priors restrict solutions to the range of a pretrained generator and search over its latent code for each measurement (Bora et al., 2017). For electromagnetic scattering, Guo et al. (2025) sample from a latent diffusion prior coupled to a physics-based likelihood, and Bao & Zang (2025) optimize in the latent space of physics-trained generative operators. Learned direct methods instead condense inversion into a single forward pass (Arridge et al., 2019), building the prior implicitly in the network (Khoo & Ying, 2019; Wei & Chen, 2019). We explicitly encode our prior: a direct network whose output space itself carries the prior.

Direct reconstruction in learned coordinates. Shao & Du (2020) predict autoencoder codes from microwave measurements. Ahmed et al. (2023) use an adversarial autoencoder's generator for phaseless acoustic data. InvLINT (Feng et al., 2022) maps integral transforms of seismic waves linearly to an embedding of the velocity map which a trained decoder maps to an image, and Auto-Linear (Feng et al., 2024) fits a linear map between separately pretrained seismic measurement

Table 3: **What the controls establish.** Transolver backbone (last row: all five); †Shapes only.

Possible source	Control	Finding	§
Head size, compute	Scratch head, matched budget	Still wins in every setting, by 17–64%	6.1
Pretraining	Same head: random decoder, image loss	Not needed for the gain	6.1
Decoder training†	Frozen, fine-tuned, jointly trained	All beat the spatial head	6.2
VAE objective	Autoencoder instead of the VAE	Within 5% on the synthetic classes	6.3
Output compression†	64 to 4,096 coordinates (pixel count)	Every width keeps $\geq$85% of the gain	6.2
Global vector†	Local 8×8 feature map, same budget	Worse than the spatial head	6.2
Low dimension alone	PCA with the same 256 coordinates	AE: 4.7–55% gain, most on Shapes	6.3
Representation†	Projecting targets onto the PCA range	Floors explain $\leq$47% of gain over PCA	6.4
Scarce paired data†	Four times more pairs, tuned rates	Still wins on every backbone and seed	6.5

and target latent spaces. We present a modular latent head and test it across operator backbones, controlling for capacity, pretraining, rank, and supervision.

Output spaces in operator learning. Operator learning has shown why the output space matters. NOMAD (Seidman et al., 2022) replaces DeepONet's linear reconstruction with a nonlinear decoder. Lanthaler et al. (2023) prove that linear reconstruction is inefficient for discontinuous outputs. L-DeepONet (Kontolati et al., 2024) and paired autoencoders (Hart et al., 2026) learn maps between latent spaces, and results on Kolmogorov widths and PCA-based operator learning explain why the benefit of nonlinearity depends on the target class (Peherstorfer, 2022; Greif & Urban, 2019; Lanthaler, 2023). Our floor analysis builds on decompositions of operator-learning error (Lanthaler et al., 2022) and on representation error for generative priors (Asim et al., 2020). There is related concurrent work on output design: Diffusion Last Layer (Park et al., 2026) learns a diffusion model over a low-rank expansion, and B2B^{-1} (Thorpe et al., 2025) inverts in a learned coefficient space.

8 Discussion and limitations

What we learned about the output space. The choice of output parametrization is a choice of prior, and is more important than research trends suggest. Linear bases work well for smooth target distributions, while nonlinear synthesis sees significant gains for sharper structures. The gains do not require pretraining, more parameters, or an undercomplete representation. Better representations of the target explain less than half of the relative gain over PCA, suggesting the ease of learning matters in this comparison. We do not isolate the roles of global mixing, decoder structure, or optimization. Our evidence suggests that the choice of prior deserves the same consideration and testing as network design in learned inverse scattering.

Limitations. Clean target data is required for pretraining nonlinear decoders, and these methods require additional time and compute. Gains from pretraining can diminish in data-starved regimes. Additionally, our gains are relative—spatial networks which are much larger and trained longer can reach similar error as latent networks. We only train and test on synthetic wavefield data at a single wavenumber with a full-aperture array of receivers. OrganAMNIST provides medical images, but not clinical wavefield measurements. We do not test methods on unseen target distributions, which is a regime where a learned output prior could potentially degrade performance. Due to volume of ablations and compute constraints, our trials are limited to 3–10 seeds, but we discuss statistical significance in Appendix A.4.

Conclusion. Our work establishes a better understanding of the regimes and mechanisms which influence the effect of output parametrizations on learned inverse scattering approaches. We design a single learned latent head, compatible with Fourier and attention-based networks, giving networks a representation of their target space which yields a more accurate reconstruction of the target contrast field. Our controlled comparisons to linear methods show when nonlinearity helps, and our gains hold without the need for pretraining or additional trainable parameters. Across our tested scattering regimes, target representation with nonlinear latents gives a more parameter-efficient method for conducting inverse reconstruction.

AI USE STATEMENT

Generative AI tools were used for writing code and running experiments, writing and validating mathematical statements and proofs, creating tables and figures, and literature discovery and citation assistance. They were also used for proofreading, fact-checking against the experiment records, and refining language in drafts. For the appendix, they were used for consolidating, organizing, and compacting notes, slides, and meeting transcripts into text and figures, and for fleshing out its language. The authors drove the scientific discovery, experimental design, and the analysis and framing of results. The authors retain responsibility for the final content and for the accuracy and completeness of this disclosure.

REPRODUCIBILITY STATEMENT

The accompanying artifact contains frozen protocols, training and scoring code, source snapshots, compact per-run and per-target results, and scripts that regenerate the principal figures and tables. Each reported evaluation identifies its target set, training seeds, checkpoint-selection procedure, and normalization. Raw forward fields, large training arrays, and model checkpoints are external inputs rather than part of the compact artifact; reproducing training therefore requires those inputs and the documented compute environment. The artifact includes source provenance and a file-integrity manifest. Figure regeneration uses stored scores and does not require new training.

ETHICS STATEMENT

OrganAMNIST images are used as contrast templates for simulated wave scattering. These experiments do not evaluate diagnosis, patient outcomes, or clinical reconstruction and should not be interpreted as clinical validation. Dataset provenance and applicable source licenses must be respected when obtaining the external inputs.

REFERENCES

Waqas W. Ahmed, Mohamed Farhat, Pai-Yen Chen, Xiangliang Zhang, and Ying Wu. A generative deep learning approach for shape recognition of arbitrary objects from phaseless acoustic scattering data. *Advanced Intelligent Systems*, 5(5):2200260, 2023. doi: 10.1002/aisy.202200260.

Simon Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019. doi: 10.1017/S0962492919000059.

Muhammad Asim, Mara Daniels, Oscar Leong, Ali Ahmed, and Paul Hand. Invertible generative models for inverse problems: Mitigating representation error and dataset bias. In *International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 399–409, 2020. URL https://proceedings.mlr.press/v119/asim20a.html.

Gang Bao and Yaohua Zang. A unified physics-informed generative operator framework for general inverse problems. *arXiv preprint arXiv:2511.03241*, 2025. IGNO.

Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed sensing using generative models. In *International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 537–546, 2017.

Shuhao Cao. Choose a transformer: Fourier or Galerkin. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24924–24940, 2021.

David Colton and Rainer Kress. *Inverse Acoustic and Electromagnetic Scattering Theory*, volume 93 of *Applied Mathematical Sciences*. Springer, 4th edition, 2019. doi: 10.1007/978-3-030-30351-8.

Heinz W. Engl, Martin Hanke, and Andreas Neubauer. *Regularization of Inverse Problems*, volume 375 of *Mathematics and Its Applications*. Kluwer Academic Publishers, 1996. doi: 10.1007/978-94-009-1740-8.

Yinan Feng, Yinpeng Chen, Shihang Feng, Peng Jin, Zicheng Liu, and Youzuo Lin. An intriguing property of geophysics inversion. In *International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 6434–6446, 2022. URL https://proceedings.mlr.press/v162/feng22a.html.

Yinan Feng, Yinpeng Chen, Peng Jin, Shihang Feng, and Youzuo Lin. Auto-linear phenomenon in subsurface imaging. In *International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13153–13174, 2024. URL https://proceedings.mlr.press/v235/feng24a.html.

Constantin Greif and Karsten Urban. Decay of the Kolmogorov N-width for wave problems. *Applied Mathematics Letters*, 96:216–222, 2019. doi: 10.1016/j.aml.2019.05.013. URL https://www.sciencedirect.com/science/article/pii/S0893965919301983.

Rui Guo, Yi Zhang, Yhonatan Kvich, Tianyao Huang, Maokun Li, and Yonina C. Eldar. Plug-and-play latent diffusion for electromagnetic inverse scattering with application to brain imaging. *arXiv preprint arXiv:2509.04860*, 2025.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In *International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569, 2023.

Emma Hart, Julianne Chung, and Matthias Chung. A paired autoencoder framework for inverse problems via Bayes risk minimization. *SIAM Journal on Scientific Computing*, 48(2):C385–C414, 2026. doi: 10.1137/25M1726935.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, 2nd edition, 2009. doi: 10.1007/978-0-387-84858-7. URL https://doi.org/10.1007/978-0-387-84858-7.

Hanyang Jiang, Yuehaw Khoo, and Haizhao Yang. Reinforced inverse scattering. *SIAM Journal on Scientific Computing*, 46(6):B884–B902, 2024. doi: 10.1137/22M153207X.

Yuehaw Khoo and Lexing Ying. SwitchNet: a neural network model for forward and inverse scattering problems. *SIAM Journal on Scientific Computing*, 41(5):A3182–A3201, 2019. doi: 10.1137/18M1222399. arXiv:1810.09675.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Katiana Kontolati, Somdatta Goswami, George Em Karniadakis, and Michael D. Shields. Learning nonlinear operators in latent spaces for real-time predictions of complex dynamics in physical systems. *Nature Communications*, 15(1):5101, 2024. doi: 10.1038/s41467-024-49411-w.

Samuel Lanthaler. Operator learning with PCA-Net: Upper and lower complexity bounds. *Journal of Machine Learning Research*, 24(318):1–67, 2023. URL https://jmlr.org/papers/v24/23-0478.html.

Samuel Lanthaler, Siddhartha Mishra, and George E. Karniadakis. Error estimates for DeepONets: A deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, 6(1):tnac001, 2022. doi: 10.1093/imatrm/tnac001. URL https://doi.org/10.1093/imatrm/tnac001.

Samuel Lanthaler, Roberto Molinaro, Patrik Hadorn, and Siddhartha Mishra. Nonlinear reconstruction for operator learning of PDEs with discontinuities. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=CrfhZAsJDsZ.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=EPPqt3uERT.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2nd edition, 2006. doi: 10.1007/978-0-387-40065-5. URL https://doi.org/10.1007/978-0-387-40065-5.

Mario Ohlberger and Stephan Rave. Reduced basis methods: success, limitations and future challenges. In *Proceedings of the Conference Algoritmy*, pp. 1–12, 2016.

Yong Zheng Ong, Zuowei Shen, and Haizhao Yang. Integral autoencoder network for discretization-invariant learning. *Journal of Machine Learning Research*, 23(286):1–45, 2022. URL https://jmlr.org/papers/v23/22-0297.html. arXiv:2203.05142.

Sungwon Park, Anthony Zhou, Hongjoong Kim, and Amir Barati Farimani. Generative neural operators through diffusion last layer. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.04139.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Benjamin Peherstorfer. Breaking the Kolmogorov barrier with nonlinear model reduction. *Notices of the American Mathematical Society*, 69(5):725–733, 2022. doi: 10.1090/noti2475. URL https://www.ams.org/notices/202205/rnoti-p725.pdf.

Jacob H. Seidman, Georgios Kissas, Paris Perdikaris, and George J. Pappas. NOMAD: Nonlinear manifold decoders for operator learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5601–5613, 2022. doi: 10.52202/068431-0405.

Wenyi Shao and Yong Du. Microwave imaging by deep learning network: Feasibility and training method. *IEEE Transactions on Antennas and Propagation*, 68(7):5626–5635, 2020. doi: 10.1109/TAP.2020.2978952.

Adam J. Thorpe, Stepan Tretiakov, Dibakar Roy Sarkar, Krishna Kumar, and Ufuk Topcu. Learning generalizable neural operators for inverse problems. *arXiv preprint arXiv:2512.18120*, 2025. doi: 10.48550/arXiv.2512.18120. URL https://arxiv.org/abs/2512.18120.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized Fourier neural operators. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=tmIiMPl4IPa.

Zhun Wei and Xudong Chen. Deep-learning schemes for full-wave nonlinear inverse scattering problems. *IEEE Transactions on Geoscience and Remote Sensing*, 57(4):1849–1860, 2019. doi: 10.1109/TGRS.2018.2869221.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for PDEs on general geometries. In *International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705, 2024.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. MedMNIST v2 – a large-scale lightweight benchmark for 2D and 3D biomedical image classification. *Scientific Data*, 10(1):41, 2023. doi: 10.1038/s41597-022-01721-8.

A EXPERIMENTAL SETUP AND IMPLEMENTATION

Tables 4 and 6 list every data set and every experiment's configuration. Throughout, y is a measurement tensor and x a contrast. We store measurements as a two-channel (real, imaginary) 64×64 array indexed by receiver and incident direction.

A.1 DATA AND MEASUREMENTS

Target classes. Contrast fields come from three synthetic target classes: Shapes (circles, rectangles, ellipses, and L-shapes), MNIST digits (LeCun et al., 1998), and low-rank radial-basis-function fields (LR-RBF, sums of 1–5 Gaussian bumps). We also evaluate OrganAMNIST organ images on the official split (Yang et al., 2023). The controlled studies add smooth random fields and an OrganAMNIST subset. Smooth random fields are stationary Gaussian Fourier fields with spectral amplitude $(1+(2\pi \cdot 0.1|\xi|)^2)^{-1}$, mapped through $(1+\tanh)/2$ and tapered to zero near the boundary. Unlike LR-RBF, they have many spatial modes. Both OrganAMNIST protocols use the same 64×64 images, scaled to $[0, 1]$ and multiplied by this taper. The controls use a smaller pool of these images, which the archived records call organ-derived. Every target is scored as a 64×64 low-pass contrast.

Measurements and noise. We compute measurements with a nonlinear Lippmann–Schwinger solver, without the Born approximation (Appendix C). For the synthetic classes, we add noise $\varepsilon \sim \mathcal{N}(0, \sigma^2 s_d^2 I)$, independently on the real and imaginary parts. Here s_d is the scalar standard deviation of the clean training measurements, and $\sigma \in \{0, 0.3, 0.6\}$ is the noise level relative to the data scale. OrganAMNIST measurements are noise-free. Training and validation fields are simulated from the raw scatterer on a 64×64 grid for the synthetic classes and a 128×128 grid for smooth fields and organ targets. Most test sets are solved on a 256×256 grid (Table 4).

Training pools. A seeded permutation of each synthetic training file holds out $\min(6000, \lfloor n/5 \rfloor)$ pairs for validation. The controlled studies train on nested subsets of 200 and 3,200 pairs from fixed pools (Table 4).

Test sets. The main test set scores all 405 final-epoch panel checkpoints, unchanged, on 2,000 new targets per class. We reject any target that duplicates a training, validation, or earlier test target. As in the training data, we simulate measurements from each raw scatterer, here on a 256×256 grid, and score reconstructions against its low-pass target. A second, low-pass test set re-solves the low-pass target itself. We use it for the resource sweeps and the robustness check in Section 5.3. Sets X, Z, and E are balanced over the four shape families and deduplicated against every earlier set. These targets are new, but design choices from earlier development remain (Appendix A.5).

A.2 BACKBONES AND OUTPUT HEADS

Backbones. The four attention models share a convolutional embedding onto a 32×32 token grid and positional encodings. Apart from width and depth, they differ only in how they mix tokens. We also use IAE-Net (Ong et al., 2022) as a sixth architecture (Appendix B.3). The embedding is a stride-two 4×4 convolution from the $2 \times 64 \times 64$ measurement array, followed by 2D sinusoidal position encodings, four attention heads, and pre-norm residual blocks (Table 5). FFNO lifts the measurement array with two normalized coordinate channels appended and keeps a 64×64 feature map. We add these channels because, without them, every FFNO operation commutes with circular shifts of the sensor grid, and the measurement-to-image map does not. The attention spatial heads reshape the token grid and upsample it with one transposed convolution and two 3×3 convolutions. FFNO's spatial head is two pointwise convolutions with a GELU.

Output heads. Every coordinate route chains a backbone, the coordinate head, and a synthesis module (Listing 1, condensed from our code without changing any operation). Synthesis modules return fields in normalized target units, $(x - \bar{x})/s_x$, so all heads share one image loss.

Table 4: Data and evaluation sets. Pairs are measurement–target pairs, and grids are forward-solver grids. Every set is scored against the 64×64 low-pass target.

Set	Size	Targets	Grid	Used for
MNIST pool	42,000 (36,000 train, 6,000 val.)	MNIST training digits	64, raw	main benchmark, its priors
LR-RBF pool	20,000 (16,000 / 4,000)	1–5 Gaussian bumps	64, raw	main benchmark, its priors
Shapes pool	4,000 (3,200 / 800)	four families	64, raw	main benchmark, priors, Shapes controls
High-data Shapes	12,800 train, 800 val.	four families	64, raw	high-data benchmark and its prior
MNIST control pool	4,000 train, 800 dev.	drawn from the MNIST files	64, raw	matched controls, prior
Smooth-field pool	4,000 train, 800 dev.; 800 test	smooth random fields	128, raw; test 256	matched controls, prior
OrganAMNIST subset	4,000 train, 800 dev.; 800 test	OrganAMNIST-64	128, raw; test 256	matched controls, prior
OrganAMNIST split	34,561 / 6,491 / 17,778	official partitions	128, raw; test 256	OrganAMNIST panel, full-split run, prior
Main test set	2,000 per class	new MNIST test digits, LR-RBF, Shapes	256, raw	main benchmark, factorial, high-data (Shapes, 1,995 after removing training overlaps)
Low-pass test set	2,000 per class	new targets	64, 128, 256; low-pass	robustness, grid refinement, resource sweeps, MNIST control confirmation
Set X	800 Shapes	new	256, raw	matched controls
Set Z	800 Shapes	new	256, raw	matched replication, width control
Set E	800 Shapes	new	256, raw	decoder regimes
Cross-solver set	1,000 circles, 1,000 ellipses	new	volume and boundary-integral	solver check (Appendix C)

Table 5: Panel backbone configurations. Trainable parameters exclude the frozen VAE decoder (2.28M) and the PCA basis.

Backbone	Token mixer / block	Width	Blocks	Trainable (spatial / PCA / VAE)
Transolver	physics attention, 32 slices	112	4	1.305M / 1.439M / 1.472M
Galerkin Transformer	softmax-free Galerkin attention	168	5	1.435M / 1.410M / 1.443M
OFormer	softmax-free Fourier-type attention	152	5	1.425M / 1.453M / 1.485M
GNOT	Galerkin attention, gated two-expert FFN	144	5	1.475M / 1.526M / 1.559M
FFNO	separable 1D spectral conv., 32 modes	80	4	1.696M / 1.954M / 1.987M

Listing 1: Coordinate head and synthesis modules (PyTorch).

```
class CoordinateHead(nn.Module):     # "CoeffHead" in the released code
    def __init__(self, c, p, feat_kind, cproj=14, hidden=256, pool=8):
        super().__init__()
        self.feat_kind = feat_kind      # "tokens" (attention) or "map" (FFNO)
        self.reduce = nn.Conv2d(c, cproj, 1)
        self.pool = nn.AdaptiveAvgPool2d(pool)
        self.fc = nn.Sequential(nn.Linear(cproj * pool * pool, hidden),
                                nn.GELU(), nn.Linear(hidden, p))

    def forward(self, h):
        if self.feat_kind == "tokens":  # (B, T, c) -> (B, c, g, g)
            B, T, c = h.shape
            g = int(round(T ** 0.5))    # 32 x 32 token grid
            h = h.transpose(1, 2).reshape(B, c, g, g)
        h = self.pool(F.gelu(self.reduce(h)))   # (B, 14, 8, 8)
        return self.fc(h.flatten(1))            # (B, p) coordinates

class LinearSynthesis(nn.Module):    # fixed PCA basis; buffers, not trained
    def __init__(self, Phi, mu, x_mean, x_std):  # Phi: (4096, p), orthonormal
        super().__init__()
        self.register_buffer("Phi", Phi / x_std)
        self.register_buffer("mu", (mu - x_mean) / x_std)

    def forward(self, z):
        return (z @ self.Phi.T + self.mu).view(-1, 64, 64)

class DecoderSynthesis(nn.Module):   # frozen pretrained VAE decoder
    def __init__(self, vae, x_mean, x_std):
        super().__init__()
        self.vae = vae.eval()
        for w in self.vae.parameters():
            w.requires_grad_(False)
        self.x_mean, self.x_std = x_mean, x_std
```

```
    @torch.no_grad()
    def encode(self, x):              # code target: posterior mean (training only)
        mu, _ = self.vae.encode(x.unsqueeze(1))
        return mu

    def forward(self, z):
        return (self.vae.decode(z).squeeze(1) - self.x_mean) / self.x_std

class CoordinateModel(nn.Module):
    def __init__(self, backbone, head, synthesis):
        super().__init__()
        self.backbone, self.head, self.synthesis = backbone, head, synthesis

    def coordinates(self, y):         # OFormer: LayerNorm(trunk tokens), see text
        return self.head(self.backbone.features(y))

    def forward(self, y):
        return self.synthesis(self.coordinates(y))
```

For the four attention backbones, `features` returns the 32×32 grid of trunk tokens. For FFNO it returns the 64×64 feature map. For OFormer, the coordinate head reads the trunk tokens after a LayerNorm, not the output of OFormer's learned-query cross-attention, which is dominated by input-independent positional structure. The spatial head keeps the cross-attention output. OFormer is therefore the one backbone where the two heads do not read the same features.

Convolutional coordinate head. The convolutional variant replaces the 1×1 projection and average pooling of Listing 1 with convolutions. Given a feature grid with c channels, it applies a 3×3 convolution to width w, stride-two 3×3 convolutions of width w_2 until the grid is 8×8, and a 1×1 convolution to k channels, each followed by a GELU. A dense map, with one hidden layer of width h when $h > 0$, then outputs the p coordinates. In the matched controls (Transolver, $c = 112$), $w = 112$, $w_2 = 64$, $k = 16$, and $h = 0$, so a single linear layer maps 1,024 features to 256 coordinates (477,952 parameters). The high-data configurations are in Appendix F (Table 23).

Decoders. The linear decoder is the mean and top principal directions of the clean training targets. The VAE encoder has three 3×3 convolution, GELU, and 2×2 max-pool stages (32, 64, and 128 channels) followed by linear maps to the mean and log-variance. The decoder mirrors it: a linear map from z to a $128 \times 8 \times 8$ tensor, three stride-two 4×4 transposed convolutions with GELU (64, 32, and 16 channels), and a final 3×3 convolution, for 2,277,633 parameters. The encoder (4,287,488 parameters) only supplies code targets during training. Where codes are standardized (Table 6), we center each coordinate and divide it by its training-target standard deviation. This affine map is absorbed into the frozen decoder, so the decoded range does not change.

A.3 Training

Main benchmark. On the synthetic classes, we train each inverse network for 120 epochs on 36,000 (MNIST), 16,000 (LR-RBF), or 3,200 (Shapes) measurement–target pairs. We use Adam (Kingma & Ba, 2015) (learning rate 5×10^{-4}, weight decay 10^{-5}, batch size 128), cosine learning-rate decay, and gradient-norm clipping at 1.0, and we evaluate the final-epoch checkpoint without selection (Algorithm 1). All five backbones stay under a common 2M trainable-parameter cap.

OrganAMNIST. We train each network for 40 epochs on the full official training split, with batch size 128, Adam, weight decay 10^{-5}, clipping at 1.0, and cosine decay. We choose its learning rate from $\{3, 5, 10\} \times 10^{-4}$ on a separate tuning seed, and keep the earliest epoch with minimum validation NMSE. Arms paired by seed share the initialized trunk and every epoch permutation.

Priors. We fit both decoders to clean training targets alone, before training any inverse network. The VAE trains for 100 epochs with batch size 256, Adam at 10^{-3} with cosine decay to 10^{-5}, and loss $\mathrm{MSE}(\mathrm{Dec}(z), x) + 10^{-4}\,\mathrm{KL}$, and we keep the checkpoint with the lowest validation loss. In the main benchmark, each training seed has its own prior, fitted on the class's training pool. The controlled studies use the main-benchmark priors for Shapes. For the other classes, they use one prior fitted on the 4,000 pool targets and selected by reconstruction error on the 800 development targets. OrganAMNIST selects one of three prior seeds on the official validation split, and the high-data prior is fitted on its 12,800 training targets.

Table 6: Experiment configurations. Coordinate heads are the pooled head of Listing 1 or the convolutional variant. Code targets are the frozen encoder's posterior means, raw or standardized. Budgets use batch size 128.

Experiment	Backbones	Coordinate head	Decoder	Code target	Training	Checkpoint	Seeds	Test set
Main benchmark (Sec. 5.1)	all five	pooled	PCA_{128}; VAE_{256}, frozen	raw (VAE only)	120 epochs	final	3	main
OrganAMNIST panel (Sec. 5.2)	all five	pooled	PCA_{256}; VAE_{256}, frozen	standardized	40 epochs	best val.	5; 10 (Transolver)	official test
Matched controls (Sec. 6.1)	Transolver	convolutional	residual, trained; VAE_{256}, frozen	none; raw	3,000 updates	final	10	X; low-pass; own sets
Full-split organ run (Sec. 6.1)	Transolver	convolutional	residual, trained; VAE_{256}, frozen	none; standardized	40 epochs	best val.	10	official test
Width and local map (Sec. 6.1)	Transolver	pooled, $p \leq 4096$; local map	residual, trained	none	3,000 updates	final	10	Z
Decoder regimes (Sec. 6.2)	Transolver	pooled	VAE_{256}: frozen, fine-tuned, or random	raw (full recipe)	3,000 updates	final	3	E
Matched rank (Secs. 6.3–6.4)	Transolver	pooled	PCA_{256}, AE_{256}, VAE_{256}, frozen	on or off	120 epochs	final	3	main
High-data (Sec. 6.5)	all five	convolutional	VAE_{256}, frozen	standardized	120 epochs	final	3	main (Shapes)
Resource sweeps (Sec. 6.5)	Transolver	pooled	VAE_{256}, frozen	raw	3,000 updates; 120 or 300 epochs	final	3	low-pass

Controlled studies. The controlled studies keep the optimizer, clipping, and cosine decay. Table 6 lists what each one changes, and Appendices D and F give their protocols.

Numerics and hardware. We implement all models in PyTorch 2.10 (Paszke et al., 2019) and train in float32 with TF32 matrix multiplication and cuDNN convolutions. Each run uses a single GPU, an NVIDIA GeForce RTX 3080 or RTX 5090 where the campaign logs its device.

Inverse training. Algorithm 1 is the training procedure for every panel checkpoint on the synthetic classes. The spatial, linear, and nonlinear routes differ only in the output module and the code weight λ, which is $\frac{1}{2}$ for the pretrained VAE route and 0 otherwise.

A.4 Evaluation and statistics

Main benchmark. For a checkpoint of route h in condition c (backbone, class, noise level) and seed s, we add noise to the m test measurements at the checkpoint's level and undo the target normalization. With v the pixel variance of the test targets,

$$R_{h,c,s} = \frac{1}{4096\, m\, v} \sum_{j=1}^{m} \|\hat{x}_j - x_j\|^2, \qquad \overline{R}_{h,c} = \frac{1}{3} \sum_{s} R_{h,c,s}, \qquad g_c = 100\Big(1 - \frac{\overline{R}_{l,c}}{\overline{R}_{s,c}}\Big).$$

The headline is the mean of g_c over the 45 synthetic conditions. A condition is won when $g_c > 0$, and a seed comparison when $R_{l,c,s} < R_{s,c,s}$. OrganAMNIST follows its own protocol, and we summarize it separately over five backbones with five paired seeds each (Table 8).

Pairing and scope of the tests. Panel comparisons are paired by seed, which fixes the data split and noise draws but not the network initialization. NMSE is a pooled ratio over objects and pixels, not a mean of per-object ratios. OrganAMNIST uses five paired seeds per backbone, and its arms share the initialized trunk and epoch permutations. We report win counts, seed-level effects, and every non-win instead of an omnibus test. The 45 conditions share classes, targets, and related backbones, so they are not independent trials. Target-level tests condition on fixed checkpoints and do not capture retraining variability (Appendix B.2).

Matched controls. The matched controls use ten paired seeds that share the trunk initialization and minibatch stream. We test each contrast with an exact two-sided sign-flip test over all 2^{10} assignments of the ten seed differences, with a Holm correction within each family of contrasts and paired-seed percentile bootstrap intervals. When all ten seeds favor one head, the raw p is 0.0020, the smallest possible. A non-significant difference is not evidence of equivalence.

Algorithm 1 Training one inverse network (synthetic-class panel)

Require: measurement–target pairs $\{(y_i, x_i)\}$, noise level σ, seed s, backbone, output route $\in$ {spatial, linear, nonlinear}, code weight λ

seed all generators with s; permute the pairs with seed s; hold out $n_{\text{val}} = \min(6000, \lfloor n/5 \rfloor)$ pairs for validation; train on at most 40,000
$s_d \leftarrow$ scalar standard deviation of the clean training measurements
$y_i \leftarrow y_i + \sigma\, s_d\, \epsilon_i$, $\epsilon_i \sim \mathcal{N}(0, I)$ on both channels, drawn once from a generator seeded with s ▷ training and validation
normalize measurements by the scalar mean and standard deviation of the noisy training measurements; set $\tilde{x}_i \leftarrow (x_i - \bar{x})/s_x$ with scalar training-target statistics
build the synthesis module (Listing 1), then the backbone and head
optimizer: Adam on all trainable parameters, learning rate 5×10^{-4}, weight decay 10^{-5}; cosine decay over 120 epochs
for epoch $= 1, \ldots, 120$ **do**
    for each minibatch b of 128 pairs in a fresh random order **do**
        if route is spatial **then**
            $\mathcal{L} \leftarrow \mathrm{MSE}\big(H_s(B(y_b)), \tilde{x}_b\big)$
        else
            $z \leftarrow R(B(y_b))$
            $\mathcal{L} \leftarrow \mathrm{MSE}\big(\mathrm{Synth}(z), \tilde{x}_b\big) + \lambda\ \mathrm{MSE}\big(z, \mu_E(x_b)\big)$ ▷ gradients pass through the frozen decoder
        end if
        backpropagate; clip the global gradient norm to 1.0; Adam step
    end for
    step the learning-rate schedule
end for
return the final-epoch weights ▷ no checkpoint selection

A.5 Disclosures

Panel seeds are 42, 123, and 456. An earlier five-seed set also included 7046 and 1744. We preregistered the main-test-set confirmation of the panel, with thresholds of at least 40 of 45 conditions, 120 of 135 seed comparisons, and a 20% mean reduction. We wrote it after the low-pass test set had already shown 44 of 45 wins, so it confirms that result on new targets rather than testing it blind. On OrganAMNIST, the ten-seed Transolver comparison was preregistered. The five-backbone extension was specified after that result was known, and it is descriptive.

Each main-benchmark prior was fitted on every clean target of its class's training pool, including the targets later held out for inverse validation. Validation is therefore a measurement holdout only, and the test targets are separate. This does not apply to the scratch heads or to the official organ split. An older Shapes development set shared all 200 of its circle targets with training, and its 4,800 pooled rows hold 4,597 distinct targets. We use it only for historical comparison.

We replaced two earlier versions of the coordinate head during development. The first used global average pooling into a 64-wide layer, which limited a 256-coordinate prediction to rank 64. The second used 4×4 pooling, which discarded much of the spatial variation in the Galerkin features. Set X was first used by an earlier matched study. Set Z, generated later and excluding set X, gives the same ordering for the matched heads (Appendix D). Historical checkpoints re-scored on the newer test sets keep their original model selection.

B Complete benchmark results

This appendix gives every value behind Section 5.

Table 7: **All main-test-set benchmark cells.** Raw-scatterer grid-256 measurements with low-pass targets, over three fixed training seeds. NMSE means and sample SDs are in units of 10^{-2}. Boldface is the lowest mean. The dagger denotes an exploratory conditional-target test with Holm correction over 45 contrasts, not training-run significance.

		Shapes			MNIST			LR-RBF		
Backbone	Head	0	0.3	0.6	0	0.3	0.6	0	0.3	0.6
Transolver	Spatial	6.07	8.47	11.63	3.35	4.03	4.95	5.86	7.34	10.13
		±0.35	±0.74	±1.77	±0.10	±0.06	±0.05	±0.05	±0.63	±0.40
	PCA 128	13.32	13.79	14.82	3.86	4.45	5.20	**2.78**	3.67	4.92
		±0.06	±0.21	±0.18	±0.03	±0.05	±0.05	±0.17	±0.30	±0.27
	VAE 256	**4.85**†	**5.30**†	**6.29**†	**2.58**†	**3.03**†	**3.78**†	2.80†	**3.51**†	**4.58**†
		±0.11	±0.05	±0.06	±0.01	±0.03	±0.04	±0.07	±0.08	±0.31
Galerkin	Spatial	11.65	11.84	12.86	3.62	3.94	4.74	5.19	6.23	7.75
		±0.56	±1.78	±1.03	±0.04	±0.09	±0.22	±0.30	±0.84	±1.94
	PCA 128	14.30	15.12	17.03	4.71	5.68	6.69	4.34	5.65	7.56
		±0.19	±0.24	±0.25	±0.03	±0.04	±0.06	±0.22	±0.45	±0.78
	VAE 256	**5.35**†	**6.35**†	**7.71**†	**2.94**†	**3.66**†	**4.55**†	**3.75**†	**4.86**†	**6.92**†
		±0.24	±0.37	±0.39	±0.04	±0.05	±0.07	±0.22	±0.17	±0.42
OFormer	Spatial	15.22	14.22	17.75	4.34	4.78	5.53	9.03	9.44	12.83
		±1.19	±1.84	±1.76	±0.17	±0.22	±0.14	±0.30	±0.35	±0.68
	PCA 128	13.39	14.25	15.55	4.30	5.38	6.42	3.66	4.57	6.02
		±0.11	±0.03	±0.18	±0.05	±0.08	±0.05	±0.11	±0.10	±0.25
	VAE 256	**5.42**†	**5.89**†	**6.88**†	**2.77**†	**3.50**†	**4.42**†	**3.26**†	**4.45**†	**5.50**†
		±0.26	±0.22	±0.22	±0.02	±0.03	±0.03	±0.15	±0.33	±0.19
GNOT	Spatial	10.83	10.53	11.77	3.50	4.05	4.76	4.40	4.67	**5.83**†
		±1.34	±0.74	±0.33	±0.11	±0.26	±0.17	±0.88	±0.24	±0.27
	PCA 128	14.33	14.89	16.24	4.55	5.58	6.62	4.01	5.00	6.57
		±0.08	±0.25	±0.35	±0.08	±0.06	±0.06	±0.40	±0.10	±0.10
	VAE 256	**5.41**†	**6.10**†	**7.59**†	**2.83**†	**3.62**†	**4.51**†	**3.18**†	**4.43**†	6.14
		±0.06	±0.20	±0.39	±0.02	±0.05	±0.07	±0.12	±0.10	±0.09
FFNO	Spatial	8.54	9.62	11.91	3.10	3.42	4.02	2.89	3.16	3.74
		±0.52	±0.35	±0.49	±0.03	±0.02	±0.01	±0.04	±0.03	±0.02
	PCA 128	11.41	11.84	12.79	3.60	3.87	4.29	1.62	1.87	**2.36**
		±0.08	±0.07	±0.04	±0.01	±0.01	±0.03	±0.21	±0.20	±0.22
	VAE 256	**4.12**†	**4.48**†	**5.27**†	**2.37**†	**2.64**†	**3.13**†	**1.50**†	**1.83**†	2.47†
		±0.07	±0.07	±0.05	±0.01	±0.01	±0.02	±0.05	±0.08	±0.05

Entries: NMSE in units of 10^{-2} (lower is better); seed SD below each mean. Noise fraction is in the column headings. Bold: lowest mean among three heads. †: lower of Spatial/VAE with two-sided paired-target $p_{\mathrm{Holm}(45)} < 0.05$. Targets: 2,000 per class; average the three fixed-checkpoint losses per target before testing. This is *conditional target-level* evidence, not significance across training runs. Test statistics, p values and target-bootstrap intervals: Table 10.

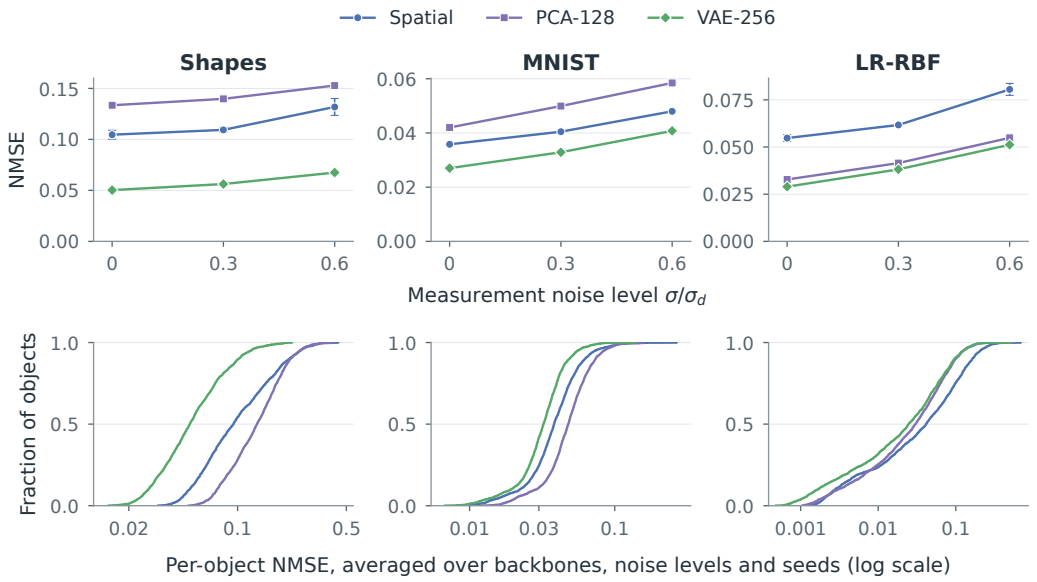

Figure 6: Main test set, averaged over the five backbones. Top: NMSE versus noise level, with backbones averaged within each seed and whiskers showing the SD over three seeds. Bottom: cumulative distributions of per-object NMSE, 2,000 objects per class, each averaged over backbones, noise levels, and seeds (log axes, different ranges). PCA is rank 128. Table 7 gives all cell values.

Table 8: **OrganAMNIST-derived scatterers, official split, five backbones.** Test NMSE on the untouched 17,778-image test split (mean±std over 5 paired seeds; validation-selected checkpoints). Win counts are paired seed comparisons. The five-backbone extension was specified after the Transolver result was known and is descriptive replication evidence; the preregistered comparison is the ten-seed Transolver row.

backbone	pixel	PCA_{256} + code	VAE_{256} + code	VAE vs pixel	VAE vs PCA
Transolver	0.0873 ±0.0020	0.0673 ±0.0040	**0.0586 ±0.0015**	-32.9% (5/5)	-13.0% (5/5)
Galerkin	0.138 ±0.0109	0.0938 ±0.0039	**0.0785 ±0.0017**	-43.1% (5/5)	-16.2% (5/5)
OFormer	0.186 ±0.0395	0.0758 ±0.0053	**0.0639 ±0.0011**	-65.7% (5/5)	-15.8% (5/5)
GNOT	0.129 ±0.0055	0.0842 ±0.0032	**0.0725 ±0.0025**	-43.8% (5/5)	-13.9% (5/5)
FFNO	0.0691 ±0.0004	0.0440 ±0.0018	**0.0409 ±0.0020**	-40.9% (5/5)	-7.0% (5/5)
backbone mean (ratio of means)	0.1219	0.0730	0.0629	-48.4%	-13.9%
Transolver, 10 seeds (preregistered)	0.0861 ±0.0026	0.0673 ±0.0046	**0.0576 ±0.0023**	-33.1% (10/10)	-14.3% (10/10)

Transolver ten-seed image-only arms (no code loss): PCA_{256} 0.0638, VAE_{256} 0.0646; the nonlinear advantage over PCA on these targets requires the code-supervised recipe. Preregistered VAE+code vs PCA+code: exact two-sided sign-flip $p = 0.0020$, paired bootstrap 95% CI $[-0.0125, -0.0070]$.

Table 9: **Selection statistics for Fig. 2.** Per-object gap $= \text{NMSE}_{\text{spatial}} - \text{NMSE}_{\text{latent}}$, averaged over seeds (three for the synthetic classes and ten for OrganAMNIST), on the same settings as the figure. The MNIST object is hand-picked as the largest-gap clean single-stroke digit (rank 17). Higher-ranked objects are malformed glyphs or multi-stroke digits.

Class	Objects	Shown index (rank)	Max gap	Median gap	Min gap	Latent better
Shapes	2,000	1645 (1)	0.235	0.0076	−0.088	73.8%
MNIST	2,000	31 (17)	0.179	0.0051	−0.006	90.6%
LR-RBF	2,000	1028 (1)	0.372	0.0145	−0.031	94.4%
OrganAMNIST	17,778	3317 (1)	0.171	0.0243	−0.138	98.7%

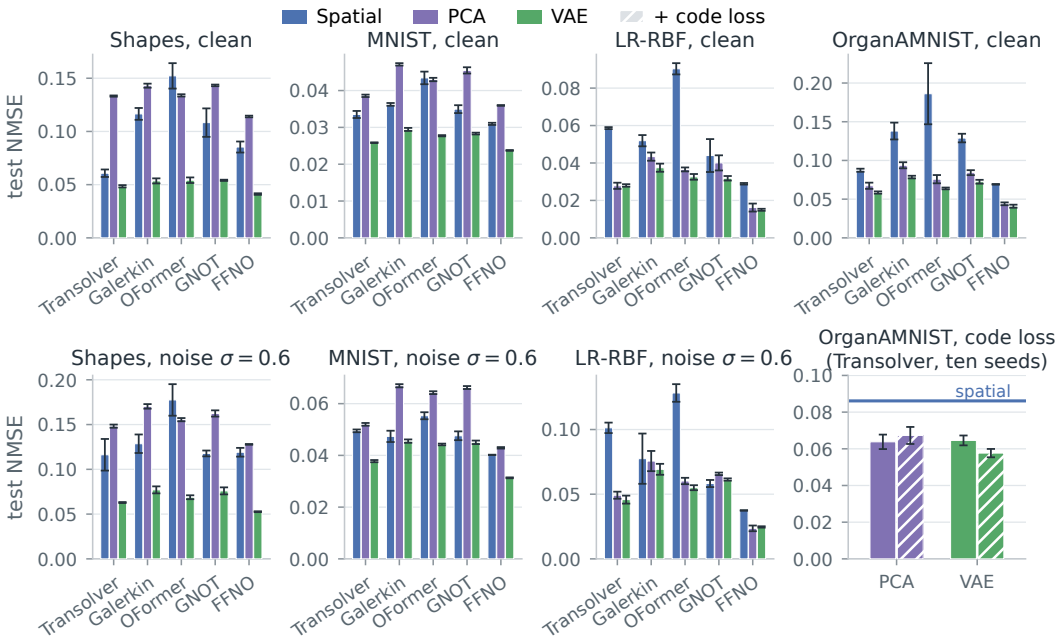

Figure 7: **Output heads across five backbones and four target classes.** Test NMSE of the spatial, PCA, and VAE heads for each backbone on the main test set, clean (top) and at noise $\sigma = 0.6$ (bottom). Bars are means and error bars sample SDs over three seeds. PCA uses 128 coordinates, and the VAE uses 256 with the code loss of Eq. (4). The VAE head beats the spatial head in every cell shown except GNOT on LR-RBF at $\sigma = 0.6$. PCA mostly loses to spatial prediction on Shapes and MNIST, and mostly wins on LR-RBF. Top right: the official OrganAMNIST split with five paired seeds, where both coordinate heads use the code loss. Bottom right: on OrganAMNIST with ten Transolver seeds, the VAE beats PCA only with the code loss (hatched), which lowers VAE error and raises PCA error. Tables 7 and 8 list every value.

B.1 Benchmark tables

B.2 Target-level tests

The seed SDs in Table 7 show how much the results vary when we retrain. Table 10 instead tests the fixed checkpoints across targets, after averaging each target's paired loss differences over the three seeds. These tests hold the checkpoints, priors, and noise draws fixed, so they say nothing about retraining. With three seeds, the smallest possible exact sign-flip p is 0.25, which is why the main text reports seed wins instead.

Table 10: Primary-cell conditional target statistics. $\Delta = 100(\text{Spatial} - \text{VAE})$. Intervals are the original nominal paired-target bootstrap CIs. p_{H45} uses all 45 target-level tests and is not a training-seed test. Two-sided one-sample t tests on paired differences, $df = 1999$. The sign convention matches Δ.

Target	Backbone	Noise	Gain (%)	Δ [95% CI]	t	p	p_{H45}
Shapes	Transolver	0	+20.2	+1.23 [+1.12, +1.33]	+23.7	1.68×10^{-109}	1.85×10^{-108}
Shapes	Transolver	0.3	+37.4	+3.16 [+3.02, +3.31]	+42.8	2.82×10^{-284}	9.86×10^{-283}
Shapes	Transolver	0.6	+45.9	+5.34 [+5.10, +5.61]	+41.4	2.75×10^{-271}	8.79×10^{-270}
Shapes	Galerkin	0	+54.1	+6.30 [+6.06, +6.53]	+52.4	$< 10^{-300}$	$< 10^{-300}$
Shapes	Galerkin	0.3	+46.4	+5.49 [+5.27, +5.71]	+48.2	$< 10^{-300}$	$< 10^{-300}$
Shapes	Galerkin	0.6	+40.1	+5.15 [+4.91, +5.40]	+41.4	4.25×10^{-271}	1.32×10^{-269}
Shapes	OFormer	0	+64.4	+9.81 [+9.43, +10.18]	+50.6	$< 10^{-300}$	$< 10^{-300}$
Shapes	OFormer	0.3	+58.6	+8.33 [+8.01, +8.64]	+52.1	$< 10^{-300}$	$< 10^{-300}$
Shapes	OFormer	0.6	+61.2	+10.87 [+10.48, +11.28]	+53.2	$< 10^{-300}$	$< 10^{-300}$
Shapes	GNOT	0	+50.1	+5.42 [+5.20, +5.64]	+48.7	$< 10^{-300}$	$< 10^{-300}$
Shapes	GNOT	0.3	+42.1	+4.44 [+4.25, +4.63]	+44.6	1.01×10^{-301}	3.63×10^{-300}
Shapes	GNOT	0.6	+35.5	+4.18 [+3.96, +4.39]	+38.2	5.17×10^{-240}	1.40×10^{-238}

Target	Backbone	Noise	Gain (%)	Δ [95% CI]	t	p	p_{H45}
Shapes	FFNO	0	+51.8	+4.42 [+4.25, +4.60]	+49.4	$< 10^{-300}$	$< 10^{-300}$
Shapes	FFNO	0.3	+53.4	+5.14 [+4.96, +5.33]	+53.9	$< 10^{-300}$	$< 10^{-300}$
Shapes	FFNO	0.6	+55.8	+6.64 [+6.43, +6.86]	+61.1	$< 10^{-300}$	$< 10^{-300}$
MNIST	Transolver	0	+22.9	+0.77 [+0.72, +0.82]	+32.0	1.89×10^{-181}	2.83×10^{-180}
MNIST	Transolver	0.3	+24.9	+1.00 [+0.95, +1.06]	+36.5	4.25×10^{-224}	1.02×10^{-222}
MNIST	Transolver	0.6	+23.7	+1.17 [+1.11, +1.23]	+37.5	8.36×10^{-234}	2.17×10^{-232}
MNIST	Galerkin	0	+18.8	+0.68 [+0.64, +0.73]	+29.9	1.56×10^{-162}	2.03×10^{-161}
MNIST	Galerkin	0.3	+7.2	+0.28 [+0.24, +0.33]	+13.1	1.22×10^{-37}	7.30×10^{-37}
MNIST	Galerkin	0.6	+3.9	+0.18 [+0.13, +0.24]	+6.9	5.07×10^{-12}	1.52×10^{-11}
MNIST	OFormer	0	+36.2	+1.57 [+1.49, +1.65]	+39.7	8.74×10^{-255}	2.54×10^{-253}
MNIST	OFormer	0.3	+26.8	+1.28 [+1.21, +1.36]	+34.0	9.90×10^{-201}	1.98×10^{-199}
MNIST	OFormer	0.6	+20.1	+1.11 [+1.04, +1.19]	+29.1	3.55×10^{-155}	4.25×10^{-154}
MNIST	GNOT	0	+19.0	+0.66 [+0.62, +0.71]	+31.4	3.32×10^{-176}	4.65×10^{-175}
MNIST	GNOT	0.3	+10.8	+0.44 [+0.39, +0.49]	+17.6	2.46×10^{-64}	1.97×10^{-63}
MNIST	GNOT	0.6	+5.3	+0.25 [+0.20, +0.31]	+9.4	9.74×10^{-21}	3.90×10^{-20}
MNIST	FFNO	0	+23.3	+0.72 [+0.69, +0.76]	+40.4	3.98×10^{-261}	1.19×10^{-259}
MNIST	FFNO	0.3	+22.9	+0.78 [+0.75, +0.82]	+42.4	1.64×10^{-280}	5.40×10^{-279}
MNIST	FFNO	0.6	+22.1	+0.89 [+0.85, +0.93]	+42.6	1.13×10^{-282}	3.85×10^{-281}
LR-RBF	Transolver	0	+52.2	+3.06 [+2.88, +3.25]	+32.4	1.85×10^{-185}	2.96×10^{-184}
LR-RBF	Transolver	0.3	+52.2	+3.83 [+3.61, +4.05]	+33.6	5.99×10^{-197}	1.14×10^{-195}
LR-RBF	Transolver	0.6	+54.8	+5.55 [+5.25, +5.85]	+35.8	2.28×10^{-217}	5.02×10^{-216}
LR-RBF	Galerkin	0	+27.8	+1.44 [+1.32, +1.58]	+22.0	3.77×10^{-96}	3.39×10^{-95}
LR-RBF	Galerkin	0.3	+22.0	+1.37 [+1.22, +1.53]	+17.4	1.42×10^{-63}	9.94×10^{-63}
LR-RBF	Galerkin	0.6	+10.6	+0.82 [+0.66, +0.98]	+10.0	8.04×10^{-23}	4.02×10^{-22}
LR-RBF	OFormer	0	+63.9	+5.77 [+5.45, +6.10]	+35.9	3.94×10^{-218}	9.06×10^{-217}
LR-RBF	OFormer	0.3	+52.9	+4.99 [+4.71, +5.29]	+33.6	1.14×10^{-196}	2.06×10^{-195}
LR-RBF	OFormer	0.6	+57.1	+7.32 [+6.88, +7.77]	+32.9	1.85×10^{-190}	3.14×10^{-189}
LR-RBF	GNOT	0	+27.6	+1.21 [+1.11, +1.32]	+22.5	8.20×10^{-100}	8.20×10^{-99}
LR-RBF	GNOT	0.3	+5.2	+0.24 [+0.14, +0.35]	+4.4	1.31×10^{-5}	1.31×10^{-5}
LR-RBF	GNOT	0.6	−5.3	−0.31 [−0.44, −0.18]	−4.6	3.99×10^{-6}	7.99×10^{-6}
LR-RBF	FFNO	0	+48.1	+1.39 [+1.32, +1.46]	+37.3	1.40×10^{-231}	3.49×10^{-230}
LR-RBF	FFNO	0.3	+42.2	+1.33 [+1.27, +1.40]	+38.6	9.44×10^{-244}	2.64×10^{-242}
LR-RBF	FFNO	0.6	+34.1	+1.28 [+1.20, +1.35]	+34.1	2.58×10^{-201}	5.41×10^{-200}

B.3 IAE-NET

We also use a compact reimplementation of the IAE-Net core (Ong et al., 2022) as a sixth backbone, with the same coordinate head on its feature map (Table 11).

Table 11: **Complete IAE-Net tier and size comparisons on the main test set.** All cells report NMSE mean $\pm$ sample SD in units of 10^{-2}, over three seeds. Small models include PCA-128, and larger tiers have no PCA runs. Gain is $100(1 - \overline{R}_{\mathrm{VAE}}/\overline{R}_{\mathrm{Spatial}})$: positive favors VAE. Sizes are spatial-model counts. Latent counts can differ, so these are not parameter-matched head comparisons. The medium IAE-Net LR-RBF latent arm has 5.05–5.35M trainable parameters across seeds. Bold marks the lowest available mean in each row.

Model / size	Target	Noise	Spatial	PCA-128	VAE-256	Gain (%)
IAE-Net						
IAE / small	LR-RBF	0	4.98 ± 0.09	2.29 ± 0.06	$\mathbf{1.99 \pm 0.13}$	+60.1
(1.45M)		0.3	5.33 ± 0.05	2.51 ± 0.04	$\mathbf{2.25 \pm 0.13}$	+57.8
		0.6	6.17 ± 0.15	3.05 ± 0.04	$\mathbf{2.80 \pm 0.01}$	+54.6
	MNIST	0	4.14 ± 0.06	3.82 ± 0.01	$\mathbf{2.52 \pm 0.01}$	+39.2
		0.3	4.45 ± 0.17	4.11 ± 0.03	$\mathbf{2.74 \pm 0.01}$	+38.4
		0.6	5.06 ± 0.22	4.62 ± 0.03	$\mathbf{3.17 \pm 0.01}$	+37.4
	Shapes	0	10.51 ± 0.32	11.24 ± 0.02	$\mathbf{4.12 \pm 0.09}$	+60.8
		0.3	11.25 ± 0.34	11.62 ± 0.02	$\mathbf{4.38 \pm 0.08}$	+61.0
		0.6	12.68 ± 0.09	12.35 ± 0.04	$\mathbf{4.96 \pm 0.07}$	+60.9
IAE / medium	LR-RBF	0	3.30 ± 0.25	—	$\mathbf{1.60 \pm 0.10}$	+51.4
(4.76M)		0.3	3.59 ± 0.27	—	$\mathbf{1.85 \pm 0.03}$	+48.5

Complete tier comparisons (continued)

Model / size	Target	Noise	Spatial	PCA-128	VAE-256	Gain (%)
		0.6	4.37 ± 0.31	—	$\mathbf{2.56 \pm 0.09}$	$+41.4$
	MNIST	0	2.88 ± 0.02	—	$\mathbf{2.35 \pm 0.02}$	$+18.5$
		0.3	3.18 ± 0.02	—	$\mathbf{2.61 \pm 0.02}$	$+18.1$
		0.6	3.74 ± 0.06	—	$\mathbf{3.06 \pm 0.02}$	$+18.4$
	Shapes	0	8.44 ± 0.08	—	$\mathbf{3.81 \pm 0.08}$	$+54.8$
		0.3	9.18 ± 0.17	—	$\mathbf{4.06 \pm 0.08}$	$+55.8$
		0.6	10.96 ± 0.27	—	$\mathbf{4.57 \pm 0.08}$	$+58.3$
IAE / large	LR-RBF	0	2.73 ± 0.16	—	$\mathbf{1.41 \pm 0.11}$	$+48.5$
(10.07M)		0.3	2.98 ± 0.18	—	$\mathbf{1.68 \pm 0.08}$	$+43.6$
		0.6	3.66 ± 0.14	—	$\mathbf{2.29 \pm 0.10}$	$+37.5$
	MNIST	0	2.69 ± 0.05	—	$\mathbf{2.30 \pm 0.00}$	$+14.6$
		0.3	3.00 ± 0.05	—	$\mathbf{2.59 \pm 0.00}$	$+13.5$
		0.6	3.61 ± 0.08	—	$\mathbf{3.15 \pm 0.02}$	$+12.5$
	Shapes	0	7.99 ± 0.06	—	$\mathbf{3.75 \pm 0.05}$	$+53.1$
		0.3	8.87 ± 0.08	—	$\mathbf{4.00 \pm 0.05}$	$+54.9$
		0.6	10.73 ± 0.07	—	$\mathbf{4.55 \pm 0.05}$	$+57.6$

C Forward simulation and numerical validation

C.1 Solvers

Volume solver. Except for the cross-solver check, all measurements come from a Lippmann–Schwinger volume discretization on $[-1, 1)^2$. With contrast x, the contrast source σ solves $(I + k^2 x G)\sigma = -k^2 x u^{\mathrm{inc}}$, where G has kernel $-\frac{i}{4} H_0^{(1)}(k|r - r'|)$, and the receiver data are $G_{\mathrm{ext}}\sigma$. The angular wavenumber is $k = 12$. For the low-pass test set, the 64×64 low-pass target is itself the scatterer. We upsample it bicubically and rebuild the operator on grids of 64, 128, and 256, and the re-solves share targets and noise draws.

Boundary-integral solver. The cross-solver check uses an independent boundary-integral solver for homogeneous objects with smooth boundaries. It solves the coupled direct system

$$\begin{bmatrix} \frac{1}{2} I - K_e & S_e \\ \frac{1}{2} I + K_i & -S_i \end{bmatrix} \begin{bmatrix} u \\ \partial_n u \end{bmatrix} = \begin{bmatrix} u^{\mathrm{inc}} \\ 0 \end{bmatrix}, \qquad u^{\mathrm{sct}} = K_e u - S_e \partial_n u,$$

where S and K are the single- and double-layer operators at the exterior (e) and interior (i) wavenumbers. We use 256 periodic nodes, trapezoidal weights, an analytic correction for the logarithmic singularity, the curvature limit on the double-layer diagonal, and a dense double-precision solve.

C.2 Grid refinement

The grid-64 and grid-256 re-solves share objects and noise draws, so their difference shows the effect of the solver grid, together with the upsampling of the scatterer (Figure 10, Table 12). On LR-RBF and MNIST, the shift is additive and similar for every head. Ratios scale with the base error and would penalize the most accurate head, so we report shifts. Shapes is the exception. There the nonlinear head shifts more than the spatial and PCA heads, for reasons we have not identified.

The main test set avoids this mismatch. Like the training data, it simulates each raw scatterer and scores against the low-pass target (Appendix A.1).

C.3 Cross-solver check

Swapping the solver family changes the error of Transolver's three heads by $1.03\times$ to $1.09\times$ and leaves their order unchanged (Table 13). By design, the volume solver sees the pixelated object, and the boundary-integral solver sees the smooth boundary.

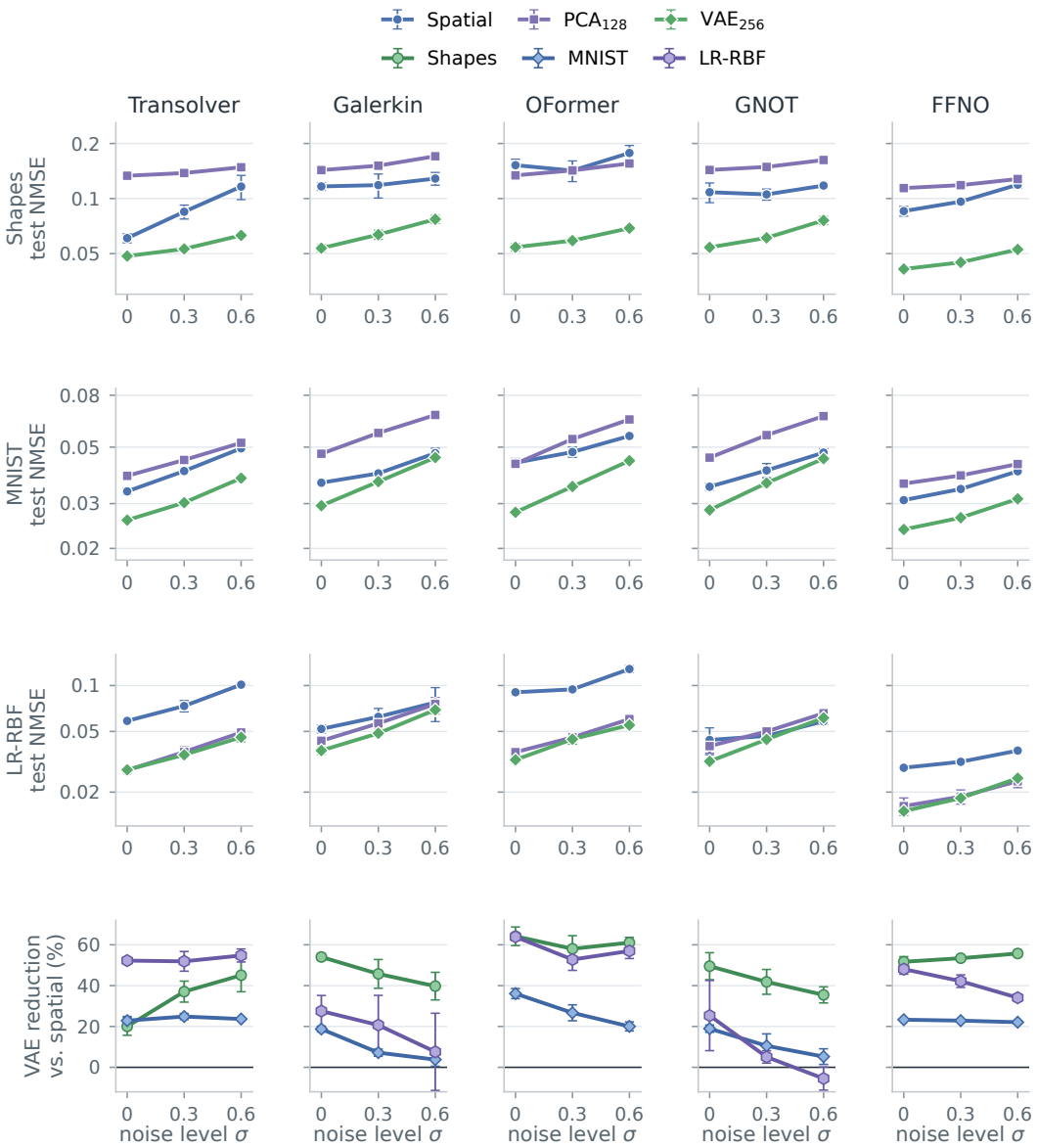

Figure 8: **Noise response of every backbone.** The per-backbone curves averaged in Figure 3a–d, one panel per backbone. Rows one to three: test NMSE against noise level for the spatial, PCA_{128}, and VAE_{256} heads on Shapes, MNIST, and LR-RBF (main test set, means and sample SDs over three seeds, log axes shared within a row). Bottom row: the VAE head's relative NMSE reduction over the spatial head per class, computed within seed. The VAE head beats the spatial head in every panel at every noise level except GNOT on LR-RBF at $\sigma = 0.6$, the single reversal of the main benchmark. Table 7 lists every value.

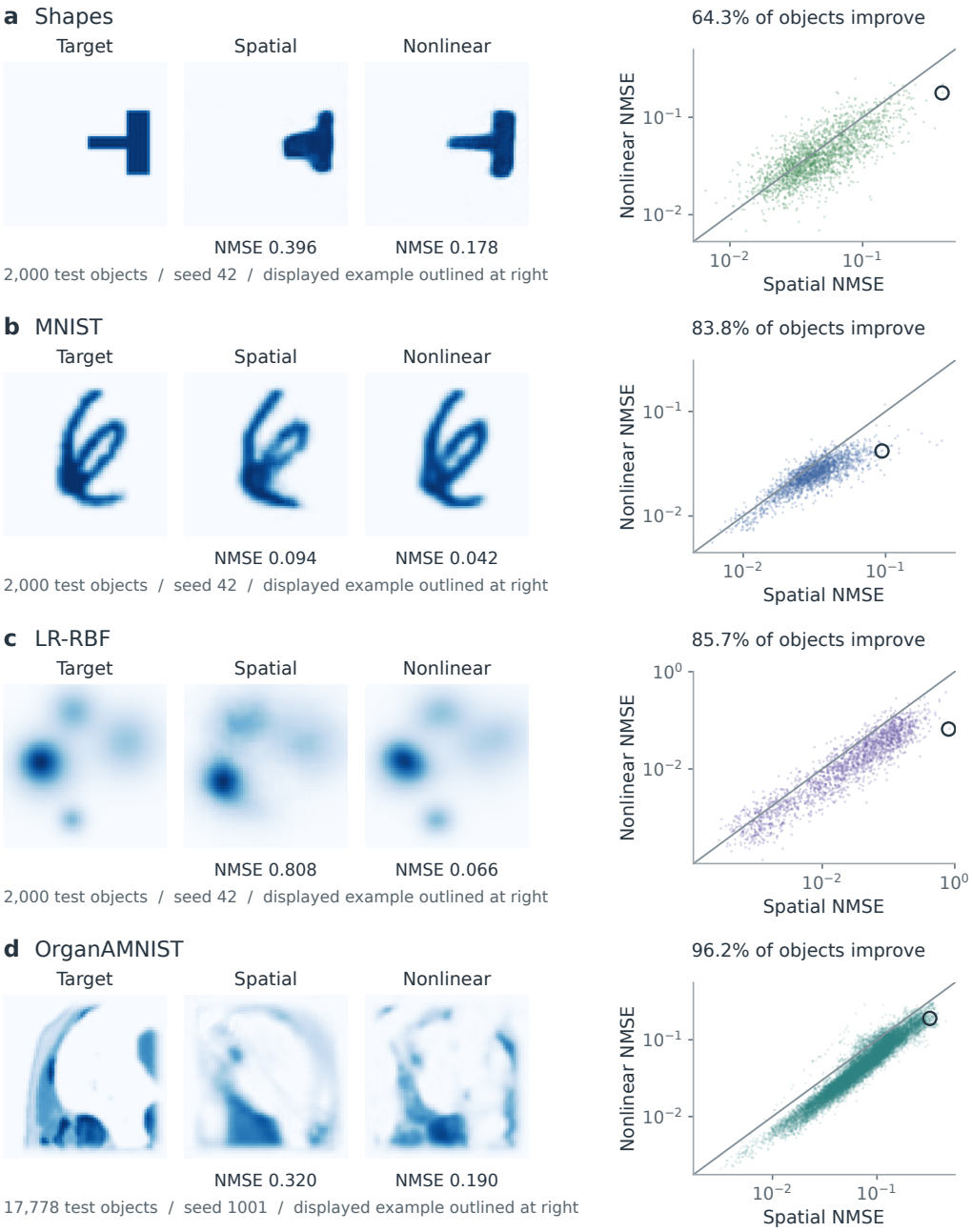

Figure 9: **Selected reconstructions and paired test errors for all four target families.** Each row uses one Transolver checkpoint (seed 42 for synthetic targets, seed 1001 for OrganAMNIST), clean measurements, and all test objects. The outlined point has exactly the errors of the displayed reconstruction. Examples were selected for large spatial-minus-nonlinear gaps, except the hand-selected MNIST example (rank 17). They are not median examples. Each image triplet shares a color scale. These single-seed win rates differ from the seed-averaged rates in Table 9, which average each object's losses over three synthetic or ten organ seeds before comparing methods.

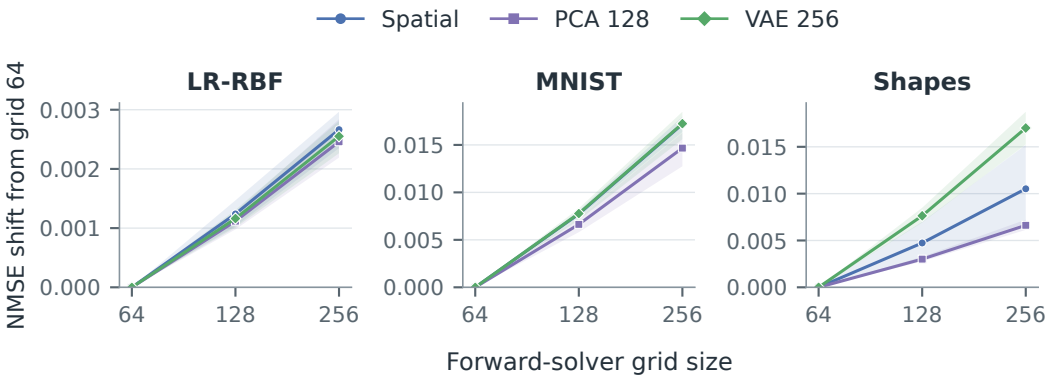

Figure 10: Change in NMSE when the low-pass test set is re-solved on a finer grid, relative to grid 64, on the same objects and noise draws. Lines average the 45 backbone, noise, and seed shifts for each head. Bands span their 10th to 90th percentiles and are not confidence intervals. Table 12 gives the values.

Table 12: **Paired grid-refinement accounting (low-pass test set).** Mean additive shift $\mathrm{NMSE}_{g256} - \mathrm{NMSE}_{g64}$ and mean ratio (parenthesized) across all backbones, noise levels, and seeds on identical fresh objects. Ratios amplify changes for lower-error methods; additive shifts are the more stable comparison.

head	LR-RBF	MNIST	shapes
pixel	$+0.0027$ ($1.05\times$)	$+0.0172$ ($1.72\times$)	$+0.0105$ ($1.12\times$)
PCA_{128}	$+0.0025$ ($1.08\times$)	$+0.0147$ ($1.43\times$)	$+0.0066$ ($1.05\times$)
VAE_{256}	$+0.0026$ ($1.09\times$)	$+0.0172$ ($2.26\times$)	$+0.0170$ ($1.44\times$)

Table 13: **Cross-solver (implementation-crime) control**, shapes domain, legacy development test set (predating the low-pass test set): the same 2,000 selection-disjoint objects measured by the training-family volume solver (LS) and by an independent boundary-integral solver (BIE; different discretization family). The complete Transolver pixel/PCA/VAE comparison changes by 1.03–$1.09\times$ and preserves the VAE ordering. Galerkin currently has pixel/PCA rows only. Circle+ellipse subset (BIE-representable); this is a scoped implementation control rather than a five-backbone replication.

backbone	head	LS (same)	BIE (cross)	ratio
transolver	pixel	0.0553	0.0584	$1.06\times$
	PCA	0.1189	0.1221	$1.03\times$
	VAE	0.0294	0.0320	$1.09\times$
galerkin	pixel	0.0804	0.0823	$1.02\times$
	PCA	0.1328	0.1360	$1.02\times$

D CONTROLLED COMPARISONS

This appendix gives the protocols behind Section 6.

D.1 MATCHED-BUDGET HEADS

Table 2 gives the main comparison, and Tables 14 and 15 give the full uncertainty estimates.

Shared trunk and training. All three heads sit on the panel's Transolver trunk (width 112, four layers, 1,175,360 parameters). Within a seed, every head starts from the same trunk initialization and sees the same nested, balanced training subset and the same minibatch stream. Only the heads are initialized separately. Each fit runs 3,000 updates of batch 128 with Adam (weight decay 10^{-5}), cosine decay to 10^{-5}, and gradient clipping at 1.0. Normalization comes from each seed's 200-pair core. The ten seeds are 1001, 2003, 3001, 4001, 5003, 6007, 7001, 8009, 9001, and 10007.

Learning rates. We choose the spatial head's learning rate per class and training size. On Shapes and the OrganAMNIST subset, we pick it from $\{3, 5, 10\} \times 10^{-4}$ by development NMSE at seed 1001, which is also a production seed. On MNIST and smooth fields, we pick it from $\{3, 10\} \times 10^{-4}$ by mean best development NMSE over the ten seeds. Both nonlinear heads use the second rule on every class. Final-step checkpoints are primary, and development-selected checkpoints give the same outcome in every cell.

On MNIST, smooth fields, and the OrganAMNIST subset, the spatial head chose the lowest rate in its grid. We therefore also trained it at 10^{-4} and 2×10^{-4} on all ten seeds, with the same trunks, subsets, and minibatch streams. At the final step, both lower rates are worse on the confirmation set in every cell. Re-selecting over the extended grid changes only smooth fields with 200 pairs, to 10^{-4}. There the scratch head wins nine of ten seeds, by 16.9% ($p = 0.0039$), and every other contrast still wins all ten. Table 2 keeps the original grid as the primary analysis.

Heads. The spatial head decodes the token grid with a width-174 convolutional stack (upsample width 78, four residual blocks at 32×32), with 2,574,763 parameters and 2,637.0M MACs. The scratch head is the convolutional coordinate head (3×3 convolution $112 \to 112$ at 32×32, stride-two 3×3 convolutions $112 \to 64 \to 64$ down to 8×8, 1×1 convolution $64 \to 16$, and Linear(1024, 256), for 477,952 parameters). It feeds a randomly initialized residual decoder: an $8 \times 8 \times 64$ seed and three stages of transposed convolution, GELU, and residual block at 16^2, 32^2, and 64^2, with 48, 88, and 168 channels. In total it has 2,575,225 parameters ($+0.02\%$) and 2,639.0M MACs ($+0.07\%$), and we train it with image loss alone. The pretrained head is the same coordinate head followed by the class's frozen VAE-256 decoder (2,277,633 parameters, fitted on 4,000 clean training targets), trained with the loss of Eq. (4) on raw code targets. It trains 1,653,312 parameters, 44% of the spatial model's 3,750,123 (Table 24).

Classes and confirmation sets. We score Shapes on set X, with set Z as a replication. MNIST is scored on the low-pass test set, and smooth fields and the OrganAMNIST subset on their own 800-object test sets. None of these sets was used for selection. The OrganAMNIST subset uses 3,200 pairs only (Table 4).

Tests. Tests follow Appendix A.4, with Holm over the seven 3,000-update cells of each nonlinear head. The ordering replicates on set Z. The spatial, scratch, and pretrained heads reach 0.4573, 0.2387, and 0.1687 with 200 pairs, and 0.0339, 0.0126, and 0.0183 with 3,200. Both nonlinear heads win all ten seeds.

Full official OrganAMNIST split. The same three heads also train on the full official split for 40 epochs at learning rate 10^{-3}, with ten paired seeds that share the trunk initialization and example order. The pretrained head uses the prior selected for the full-split study.

D.2 Coordinate width and the local-map control

Setup. These controls use the pooled coordinate head of Listing 1 on Shapes, with the protocol above (image loss alone, clean measurements, ten paired seeds) and set Z as primary. The head predicts 64, 256, or 4,096 coordinates, and 4,096 is the number of pixels. Its projection has $\max(14, \lceil p/64 \rceil)$ channels, its hidden layer $\max(256, p)$ units, and its decoder starts with Linear(p, $64 \cdot 8 \cdot 8$). The local-map decoder removes the global vector. Two stride-two 3×3 convolutions ($112 \to 872 \to 64$) produce an $8 \times 8 \times 64$ map that feeds the same upsampling stages at widths 80, 96, and 160, with no flattening, pooling, or dense layer (2,577,081 parameters, $+0.09\%$, and 2,638.2M MACs, $+0.05\%$). We choose learning rates per width and training size, and both new widths reach a training MSE below 0.05 over the last 100 updates at 3,200 pairs.

Result. Every width stays far below the spatial head and keeps at least 85% of the width-256 head's advantage, so a strictly undercomplete vector is not needed for the gain (Figure 13). Width 256 is best on both sets and at both sizes. With Holm over the four set-Z contrasts, width 4,096 is worse with 200 pairs (11.9% higher NMSE, $p_{\mathrm{H}} = 0.0078$), and width 64 is worse with 3,200 (10.3%, $p_{\mathrm{H}} = 0.0078$). The local map, which removes the global coordinate vector, is worse than the spatial head at both sizes.

Table 14: **Matched scratch-head replication: complete results.** Independently trained Transolver backbones, paired initializations and data streams, image loss only, and 3,000 updates. Head parameters and MACs match within 0.1%. All eight contrasts are shown.

Target	n_{train}	Spatial	Synthesis	Gain (%)	Δ [95% CI]	p_{H7}
Shapes	200	44.18 $\pm$3.84	**23.75** $\pm$2.96	+46.2	+20.43 [+17.82, +22.60]	0.0137
Shapes	3,200	3.52 $\pm$0.37	**1.27** $\pm$0.20	+64.1	+2.26 [+2.05, +2.43]	0.0137
MNIST	200	52.67 $\pm$1.29	**36.54** $\pm$3.52	+30.6	+16.13 [+13.92, +18.09]	0.0137
MNIST	3,200	7.64 $\pm$0.18	**5.36** $\pm$0.57	+29.8	+2.28 [+1.88, +2.58]	0.0137
Smooth fields	200	87.80 $\pm$1.44	**73.12** $\pm$7.05	+16.7	+14.68 [+10.36, +18.59]	0.0137
Smooth fields	3,200	20.32 $\pm$0.33	**12.23** $\pm$0.54	+39.8	+8.09 [+7.71, +8.44]	0.0137
OrganAMNIST	3,200	13.60 $\pm$0.21	**9.74** $\pm$0.61	+28.4	+3.87 [+3.50, +4.24]	0.0137
OrganAMNIST	34,561	7.29 $\pm$0.32	**4.71** $\pm$1.21	+35.3	+2.58 [+1.78, +3.30]	0.0039*

NMSE and Δ in units of 10^{-2}; SD below means. $\Delta = \text{Spatial} - \text{Synthesis}$; positive gain favors the nonlinear head. Clean measurements; Transolver trunk; 3,000 updates. Scratch nonlinear head: the convolutional coordinate head and a residual decoder trained from random initialization with image loss alone (Appendix A); head parameters and MACs within 0.1% of the spatial head. Ten paired training seeds per row. Confirmation sets: shapes set X (800 objects); MNIST, the low-pass test set at grid 256 (2,000); smooth fields, the extension split (800); OrganAMNIST subset, an 800-object set generated after checkpoint freeze. CIs: paired-seed bootstrap, conditional on fixed confirmation sets and selected learning rates. p_{H7}: exact two-sided paired-seed sign-flip tests, Holm-corrected over the seven contrasts. Every row above the rule has raw $p = 0.0020$, the smallest attainable with ten seeds. *OrganAMNIST: the full official split (34,561 training pairs, 17,778 test objects), 40 epochs at learning rate 10^{-3}, validation-selected checkpoints; 9 of 10 seeds won, raw $p = 0.0039$, Holm-corrected over its two contrasts. CIs are nominal, not simultaneous.

Table 15: **Frozen pretrained decoder behind the same coordinate head: complete results.** Same trunks, paired initializations, data streams, confirmation sets, and 3,000 updates as Table 14.

Target	n_{train}	Spatial	Pretrained	Gain (%)	Δ [95% CI]	p_{H7}
Shapes	200	44.18 $\pm$3.84	**16.92** $\pm$0.63	+61.7	+27.26 [+24.90, +29.53]	0.0137
Shapes	3,200	3.52 $\pm$0.37	**1.79** $\pm$0.02	+49.3	+1.74 [+1.53, +1.96]	0.0137
MNIST	200	52.67 $\pm$1.29	**19.80** $\pm$1.15	+62.4	+32.88 [+31.96, +33.78]	0.0137
MNIST	3,200	7.64 $\pm$0.18	**4.42** $\pm$0.06	+42.1	+3.21 [+3.10, +3.32]	0.0137
Smooth fields	200	87.80 $\pm$1.44	**44.74** $\pm$1.25	+49.0	+43.05 [+41.90, +44.06]	0.0137
Smooth fields	3,200	20.32 $\pm$0.33	**13.45** $\pm$0.26	+33.8	+6.87 [+6.56, +7.12]	0.0137
OrganAMNIST	3,200	13.60 $\pm$0.21	**7.47** $\pm$0.19	+45.1	+6.13 [+5.96, +6.29]	0.0137
OrganAMNIST	34,561	7.29 $\pm$0.32	**3.43** $\pm$0.06	+53.0	+3.87 [+3.70, +4.07]	0.0039*

NMSE and Δ in units of 10^{-2}; SD below means. $\Delta = \text{Spatial} - \text{Pretrained}$; positive gain favors the nonlinear head. Clean measurements; Transolver trunk; 3,000 updates. Pretrained nonlinear head: the same convolutional coordinate head followed by the class's frozen pretrained VAE decoder (2.28M parameters, fitted on 4,000 clean training targets), trained with the loss of Eq. (4); 44% of the spatial model's trainable parameters. Ten paired training seeds per row. Confirmation sets: shapes set X (800 objects); MNIST, the low-pass test set at grid 256 (2,000); smooth fields, the extension split (800); OrganAMNIST subset, an 800-object set generated after checkpoint freeze. CIs: paired-seed bootstrap, conditional on fixed confirmation sets and selected learning rates. p_{H7}: exact two-sided paired-seed sign-flip tests, Holm-corrected over the seven contrasts. Every row above the rule has raw $p = 0.0020$, the smallest attainable with ten seeds. *OrganAMNIST: the full official split (34,561 training pairs, 17,778 test objects), 40 epochs at learning rate 10^{-3}, validation-selected checkpoints; 10 of 10 seeds won, raw $p = 0.0020$, Holm-corrected over its two contrasts. CIs are nominal, not simultaneous.

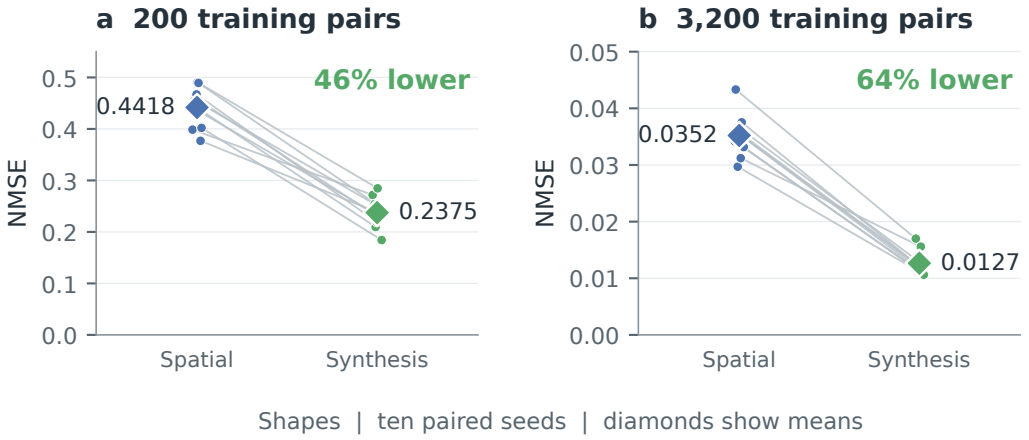

Figure 11: The ten paired Shapes seeds behind the scratch head of Table 2, on set X. Each line connects one seed's spatial and scratch-synthesis results, and diamonds mark means. Image loss alone, no pretraining or code supervision, and head parameters and MACs within 0.1%. Final-step checkpoints.

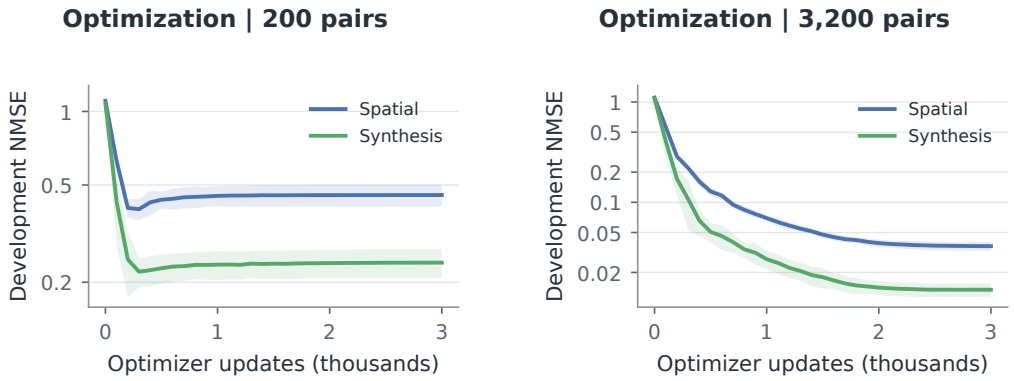

Figure 12: Development NMSE during training for the spatial and scratch heads of Table 2 on Shapes, with 200 and 3,200 pairs. Curves are unsmoothed means over ten paired seeds, evaluated every 100 updates on the 800-object development split. Bands show one SD. These are optimization diagnostics, not test learning curves.

D.3 Decoder regimes

Setup. The decoder-regime study of Section 6.2 uses Shapes, the Transolver trunk, and the pooled coordinate head (Table 16). Its scratch row is the matched head above. All regimes use nested, balanced subsets, paired initialization and minibatch streams at each training size, normalization from each seed's 200-object core, 3,000 updates of batch 128, and seeds 42, 123, and 456. The spatial, frozen, and fine-tuned regimes use learning rate 3×10^{-4}, and the fine-tuned decoder trains at the same rate. The random joint decoder uses 5×10^{-4} and adds its 2.28M decoder parameters to the trainable count. Code targets are raw posterior means. We score on set E, which was never used for selection.

Result. Every nonlinear regime beats the spatial head at both training sizes (Table 17; Figure 14b), but none is best at both. Fine-tuning wins with 3,200 pairs, and the full frozen recipe wins with 200. The 200-pair ordering also depends on when we stop. Replaying the runs to their validation-selected steps reorders them, so we do not read much into the ordering. At the common learning rate of 3×10^{-4}, the random joint decoder still stays below the spatial head.

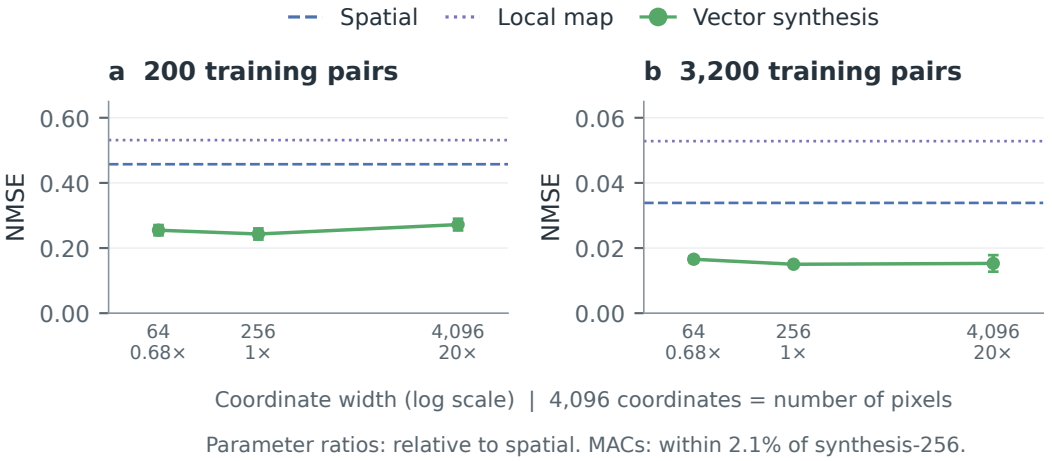

Figure 13: Coordinate width on Shapes, set Z, with 200 and 3,200 pairs. Vector synthesis through the pooled coordinate head at widths 64, 256, and 4,096 (points: mean and SD over ten paired seeds) against the spatial head and the local-map decoder (lines: means). Scratch training with image loss alone, clean measurements. Head MACs are within 2.1% of the width-256 head, and head parameters are 0.68, 1, and 20 times the spatial head's, and width 4,096 equals the pixel count. The pooled head makes these values differ from the convolutional scratch head of Table 2.

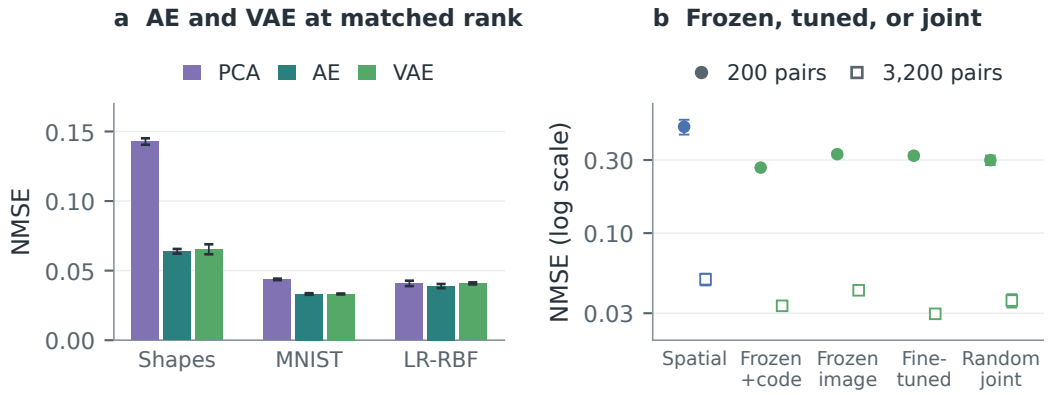

Figure 14: Several realizations of learned-coordinate prediction, using Transolver in both panels. (a) At the same 256-coordinate width and with image loss alone, compare PCA, AE, and VAE on all three synthetic target classes of the main test set. Noise 0 and 0.6 are averaged within each of three seed indices. Bars and whiskers show mean and sample SD. Both nonlinear families improve strongly on Shapes and MNIST. On LR-RBF the benefit is small or absent. (b) All five decoder-state arms at both 200 and 3,200 pairs on the separate Shapes E confirmation set (800 objects, clean, Transolver): spatial, pretrained-frozen with code loss, pretrained-frozen with image loss, fine-tuned pretrained, and random jointly trained. Points are final-step means and sample SDs over three seeds after 3,000 updates. All four nonlinear configurations improve on spatial at both budgets. These state experiments have different trainable resources and the random-joint arm uses a different learning rate. Table 2 separately tests matched scratch budgets. The main test set and set E are not pooled. Full protocols and values: Table 19 and Appendix D.3.

Table 16: Decoder regimes (Transolver trunk, Shapes). The first four use the pooled coordinate head, and scratch synthesis uses the convolutional head. "Image" is the normalized image MSE, and "code" adds $\frac{1}{2}\,\mathrm{MSE}(z, \mu_E(x))$. The encoder is never trained.

Regime	Decoder initialization	Decoder trained	Loss	Decoder learning rate
Frozen, full recipe	pretrained VAE	no	image + code	–
Frozen, image only	pretrained VAE	no	image	–
Fine-tuned	pretrained VAE	yes	image	base rate[†]
Random joint	random, VAE architecture	yes	image	base rate
Scratch synthesis	random, residual decoder[*]	yes	image	see App. D.1

[†]Ratio to the base rate chosen from $\{0.01, 0.1, 1\}$, which selected 1. [*]Linear map to $64 \times 8 \times 8$, then three stages of stride-two transposed convolution and a residual block (48, 88, and 168 channels), and a 3×3 output convolution, behind the convolutional coordinate head. Its head parameters are within 0.02% and its MACs within 0.07% of the matched spatial head (Appendix D).

Table 17: Decoder regimes on set E. Final-step NMSE, mean $\pm$ SD over three seeds. The last column replays each 200-pair run to the step selected by its original validation trajectory.

Regime	200 pairs	3,200 pairs	200 pairs, replayed
Spatial	0.495 ± 0.055	0.0500 ± 0.0044	0.407 ± 0.029
Frozen, full recipe	$\mathbf{0.268} \pm 0.010$	0.0335 ± 0.0018	0.258 ± 0.014
Frozen, image only	0.328 ± 0.014	0.0424 ± 0.0028	0.284 ± 0.022
Fine-tuned	0.320 ± 0.015	$\mathbf{0.0297} \pm 0.0011$	$\mathbf{0.249} \pm 0.014$
Random joint	0.300 ± 0.021	0.0363 ± 0.0036	0.257 ± 0.023
Random joint, learning rate 3×10^{-4}	0.312 ± 0.021	0.0423 ± 0.0043	–

E REPRESENTATION, PREDICTION, AND TRAINING IN LEARNED COORDINATES

Our controlled experiments separate three questions. How well can a decoder represent a target? How accurately can its coordinates be predicted from the measurements? And how does the training objective weight prediction errors? This appendix gives simple identities and bounds that connect these questions. They hold for pretrained and jointly trained decoders once their weights are fixed. We build on the distinction between representation error in generative inverse models (Asim et al., 2020) and the parts of operator-learning error (Lanthaler et al., 2022). These are accounting identities, not recovery or generalization guarantees for the scattering problem.

Notation and scope. Let $X \in \mathbb{R}^d$ be the target field and Y the measurements, with $\mathbb{E}\|X\|^2 < \infty$. Write $h(Y) = D(g(Y))$, where g includes the backbone and coordinate readout. All reconstruction risks use the same normalizer $V = dv > 0$, with v the pixel variance of the scored targets, so $R(h) = \mathbb{E}\|X - h(Y)\|^2/V$. We condition on the fitted models, bases, and normalizers, and assume the random quantities are measurable with finite second moments. The algebraic statements also hold as averages over a fixed evaluation set. Section E.3 instead concerns the population distribution of a fresh measurement–target pair, independent of fitting and selection. Conditioning on a trained decoder does not give a generalization bound for the procedure that learned it.

E.1 HOW MUCH OF A MEASURED GAIN CAN REPRESENTATION EXPLAIN?

For a nonempty synthesis range S, define its representation floor and a predictor's excess above that floor by

$$F_S = \mathbb{E}\ \mathrm{dist}(X, S)^2/V, \qquad E_S = R(h_S) - F_S \geq 0, \qquad h_S(Y) \in S.$$

The excess E_S includes measurement ambiguity, the restricted predictor class, finite training data, and optimization. It is not only optimization error.

Proposition 1 (Bounding the gain beyond representation)**.** *Compare an affine PCA synthesis range A and a nonlinear decoder range N. Let $G = R(h_A) - R(h_N)$, and let U_N be the error of any achieved reconstruction of the targets in N, such as an encoder–decoder roundtrip. Then*

$$G - F_A \ \leq\ E_A - E_N \ \leq\ G - F_A + U_N. \tag{5}$$

If $G > F_A$, at least the fraction $1 - F_A/G$ of the observed advantage remains after subtracting the largest possible representation-floor advantage.

Proof. Pointwise distance minimization gives $0 \leq F_N \leq U_N$. Subtracting the two risk decompositions yields $E_A - E_N = G - F_A + F_N$, proving both inequalities. The fractional statement follows by division by $G > F_A \geq 0$. □

For the actual fitted range $A = \mu + \mathrm{span}(\Phi)$, with $\Phi^\mathsf{T}\Phi = I$, the PCA projection identity (Hastie et al., 2009, Section 14.5.1) gives

$$F_A = \mathbb{E}\big\|(I - \Phi\Phi^\mathsf{T})(X - \mu)\big\|^2/V. \tag{6}$$

This is the projection error of the fitted basis on the evaluation targets. The smallest rank-p tail from refitting on those targets can be smaller, so it cannot replace F_A in the lower bound.

Application to the Shapes controls. Table 18 applies the bound to the image-only, 256-coordinate Shapes comparison on the low-pass test set. The risks and projection floors use the same targets, normalizers, and weights across three seeds. We apply Eq. (5) to each seed and average. At least 53–56% of the improvement over PCA remains after allowing the largest possible representation-floor benefit. These are deterministic bounds on the reported means, not confidence intervals. They concern the comparison with PCA, not the headline improvement over spatial prediction. An encoder roundtrip or an optimized code only gives an upper bound on F_N.

Table 18: Representation-floor accounting on the matched-width Shapes controls. G and F_A are seed-averaged normalized errors.

Nonlinear head	Noise	G	F_A	Lower bound $(G - F_A)/G$
AE	0.0	0.08155	0.03603	55.8%
VAE	0.0	0.07928	0.03603	54.6%
AE	0.6	0.07805	0.03603	53.8%
VAE	0.6	0.07687	0.03603	53.1%

E.2 From reconstruction error to inverse-prediction error

To compare pretrained and scratch decoders without a true nonlinear projection, we choose any measurable reference code $z_*(X)$. It can be an encoder output, or a code found by fitting the target with the frozen decoder. Define

$$a = D(z_*(X)) - X, \qquad b = D(g(Y)) - D(z_*(X)), \qquad U = \mathbb{E}\|a\|^2/V, \qquad Q = \mathbb{E}\|b\|^2/V.$$

Here U is an achieved reconstruction error, and Q measures the gap between predicted and reference codes *after decoding*.

Proposition 2 (Decoded prediction and reconstruction)**.** *The inverse risk satisfies*

$$R(h) = U + Q + 2\mathbb{E}\langle a, b\rangle/V, \qquad (\sqrt{U} - \sqrt{Q})^2 \le R(h) \le (\sqrt{U} + \sqrt{Q})^2. \tag{7}$$

If $\|D(g(Y)) - D(z_*(X))\| \le L\|g(Y) - z_*(X)\|$ almost surely for a finite constant L, then

$$\sqrt{R(h)} \le \sqrt{U} + L\sqrt{\mathbb{E}\|g(Y) - z_*(X)\|^2/V}. \tag{8}$$

Proof. Expand $\|a+b\|^2$ and apply Cauchy–Schwarz to its expected cross-term. The assumed decoder bound gives $Q \le L^2\mathbb{E}\|g(Y) - z_*(X)\|^2/V$; substitution into the upper bound proves Eq. (8). □

For PCA reference codes that give the orthogonal projection, a is orthogonal to b, so the cross-term is zero. For a nonlinear decoder it generally is not, even for nearest-point reference codes, because tangent-space orthogonality need not hold for the finite chord b. So subtracting an autoencoder's roundtrip error from the inverse error does not isolate prediction or optimization error. The decoded gap Q also avoids a coordinate-scale ambiguity. An invertible affine change $\widetilde{z} = Mz + c$, absorbed into the decoder, keeps U, Q, and the predictions, but changes the Euclidean code error to $(z - z_*)^\mathsf{T} M^\mathsf{T} M(z - z_*)$.

These identities apply to the final decoder of a scratch model as well as a pretrained one. Reference codes fitted with the true test targets are oracle diagnostics, not inputs the inverse predictor has. We do not estimate a global L, and we do not use this bound to claim that one decoder is better conditioned. The gains at both 64 and 4,096 coordinates also do not give an error bound that depends on latent width alone.

E.3 Why clean-target reconstruction and inverse training can disagree

A clean-target decoder is fitted to represent X. Under squared-error inverse prediction, the best unrestricted answer is instead the conditional mean $m(Y) = \mathbb{E}[X \mid Y]$. The following is the standard conditional-mean identity for squared loss (Hastie et al., 2009, Section 2.4), applied to a fixed synthesis range.

Proposition 3 (Population risk with a fixed output range)**.** *Let $R_{\mathrm{B}} = \mathbb{E}\|X - m(Y)\|^2/V$. Every square-integrable predictor $h(Y)$ satisfies*

$$R(h) = R_{\mathrm{B}} + \mathbb{E}\|m(Y) - h(Y)\|^2/V. \tag{9}$$

For S-valued predictors this implies

$$R(h) \ge R_{\mathrm{B}} + \mathbb{E}\ \mathrm{dist}(m(Y), S)^2/V. \tag{10}$$

If a measurable nearest-point map P_S exists and $P_S(m(Y))$ is square-integrable, equality is attained by $h(Y) = P_S(m(Y))$ among all measurable S-valued predictors.

Proof. Expand $X - h(Y) = (X - m(Y)) + (m(Y) - h(Y))$. Conditional expectation given Y annuls the cross-term because $\mathbb{E}[X - m(Y) \mid Y] = 0$. Distance minimization gives the inequality, and the stipulated projection attains it. □

The equality holds over all measurable range-valued predictors. A finite backbone and coordinate head need not reach the minimizer. Unlike F_S, which measures distance from individual targets, the second term in Eq. (10) measures distance from their conditional means, and the two need not agree. For example, take $V = 1$, let X be $(1, 0)$ or $(-1, 0)$ with equal probability, and let Y carry no information about X. The unit circle S represents both targets exactly, so $F_S = 0$. But $m(Y) = 0$ and $R_{\mathrm{B}} = 1$. Every circle-valued predictor has risk 2, while the unrestricted zero predictor has risk 1.

This is why decoder reconstruction quality alone cannot rank inverse predictors. Pretraining fits clean-target reconstruction, while joint image-loss training can adapt the decoder and predictor to the inverse objective. The gains in Sections 6.1 and 6.2 are consistent with different mixes of representation and prediction error, and the proposition does not say which mix produces a given gain. Restricting the output to a decoder range also cannot beat the unrestricted Bayes risk, so its value has to be measured with real, finite-data predictors.

E.4 What code supervision changes in PCA and nonlinear decoding

Section 6.4 finds that code supervision affects PCA and nonlinear heads differently. What it does depends on the coordinates. Let $V_{\mathrm{tr}} = ds_x^2$, where s_x is the training-target standard deviation in Eq. (4). This training normalizer need not equal the evaluation normalizer V. For PCA write

$$D(z) = \mu + \Phi S z, \qquad \Phi^{\mathsf{T}}\Phi = I, \qquad S = \mathrm{diag}(s_1, \ldots, s_p), \quad s_j > 0,$$
$$z_*(X) = S^{-1}\Phi^{\mathsf{T}}(X - \mu).$$

Raw PCA coordinates use $S = I$. For standardized coordinates, s_j are the positive coefficient scales used in our code, including its numerical lower bound, and need not equal the population PCA eigenvalue square roots. Positive rescaling leaves the synthesis range unchanged.

Proposition 4 (PCA code supervision is coordinate reweighting)**.** *For $\lambda \geq 0$, the image-plus-code objective has the exact form*

$$\begin{aligned}\mathcal{L}(z, X) &= \frac{\|D(z) - X\|^2}{V_{\mathrm{tr}}} + \frac{\lambda}{p}\|z - z_*(X)\|^2 \\ &= \frac{\|(I - \Phi\Phi^{\mathsf{T}})(X - \mu)\|^2}{V_{\mathrm{tr}}} + \sum_{j=1}^{p}\left(\frac{s_j^2}{V_{\mathrm{tr}}} + \frac{\lambda}{p}\right)(z_j - z_{*,j}(X))^2. \end{aligned} \tag{11}$$

Among unrestricted square-integrable coordinate predictors $z = g(Y)$, its population minimizer is $g(Y) = \mathbb{E}[z_*(X) \mid Y]$ for every $\lambda \geq 0$.

Proof. The PCA projection residual is orthogonal to $\Phi S(z - z_*)$, whose squared norm is $\sum_j s_j^2(z_j - z_{*,j})^2$. This proves the identity. Its weights are positive constants and its residual is independent of g. Conditional squared-error minimization in each coordinate gives the stated minimizer. □

For raw coordinates, adding code loss rescales the predictor-dependent image loss by $1 + \lambda V_{\mathrm{tr}}/p$ and does not change its minimizers. That does not mean training follows the same path under a fixed optimizer, clipping, or regularization. For standardized coordinates, coordinate j's weight grows by a factor $1 + \lambda V_{\mathrm{tr}}/(ps_j^2)$, so small-scale directions gain the most. The same code-loss coefficient can therefore weight errors very differently without changing what the basis can represent. With a restricted predictor, finite data, or regularization, this can change the fitted solution even though the unrestricted optimum stays the same. This calculation helps interpret the PCA supervision controls, but it does not by itself explain their higher test error.

Local geometry for a nonlinear decoder. For a twice-differentiable decoder and a fixed reference code $e(X)$, take the half-scaled training loss

$$\ell(z, X) = \frac{\|D(z) - X\|^2}{2V_{\mathrm{tr}}} + \frac{\lambda}{2p}\|z - e(X)\|^2.$$

Write $r = D(z) - X$ and $J_D = \partial D/\partial z$. Direct differentiation gives

$$\nabla_z \ell = J_D^\mathsf{T} r/V_{\mathrm{tr}} + \lambda(z - e(X))/p,$$
$$\nabla_z^2 \ell = \underbrace{J_D^\mathsf{T} J_D/V_{\mathrm{tr}} + \lambda I/p}_{\text{Gauss–Newton matrix}} + \frac{1}{V_{\mathrm{tr}}} \sum_{i=1}^{d} r_i \nabla_z^2 D_i(z). \tag{12}$$

This is the standard nonlinear least-squares Hessian decomposition (Nocedal & Wright, 2006, Chapter 10). For PCA the residual-curvature term vanishes and $J_D^\mathsf{T} J_D = S^2$. For a nonlinear decoder both terms can change with the code. A frozen decoder still shapes the gradients sent to the coordinate predictor through J_D^T, and joint training changes the decoder as well. Adding code loss shifts both the Gauss–Newton matrix and the Hessian at a fixed code. It does not make the full training objective convex or guarantee lower test error, and Proposition 4 has no general counterpart for nonlinear D. These identities show why parameter counts, coordinate width, and representation floors alone do not determine training behavior. The advantages we measure remain empirical findings, and we do not claim faster convergence or a single mechanism.

E.5 Supporting diagnostics

Each diagnostic below supports one of the statements above. The factorial is a planned control. The other checks are post hoc or exploratory, and none of them enters the main benchmark.

Matched-rank and code-supervision factorial. Figure 15 and Table 19 give the factorial behind Sections 6.3 and 6.4. It uses Transolver, 256 coordinates, 120 epochs, three seeds, and the main test set at noise 0 and 0.6. Code-supervised arms use standardized code targets, and the recipe row uses the raw codes of Eq. (4). With image loss alone, the AE and the VAE each beat PCA on Shapes and MNIST in all six seed–noise comparisons. These six comparisons are not independent replications. Table 18 applies Proposition 1 to the image-only Shapes arms, scored on the low-pass test set. Proposition 4 shows that code loss only reweights PCA coordinates and leaves the range unchanged. Any change in PCA error under code loss therefore comes from the fitted predictor, not from representation.

A dense affine output layer also performs like PCA on clean Shapes (NMSE 0.139, against 0.138 for PCA and 0.057–0.059 for the nonlinear heads, on the low-pass test set). So compactness alone does not explain the comparison.

Table 19: Complete available main-test-set ingredient means and sample seed SDs (NMSE, units of 10^{-2}). Dashes indicate unrecorded arms, not zero errors. Three training seeds per available cell. The spatial controls have unequal resources. Spatial query: the learned-query spatial head of the factorial study, not the spatial head of the panel. Recipe: VAE-256 with code loss on raw (unstandardized) codes.

	Shapes		MNIST		LR-RBF	
Arm / objective	0	0.6	0	0.6	0	0.6
Spatial query	6.59 ±0.69	9.29 ±0.59	—	—	—	—
PCA 256 / image	13.65 ±0.27	14.91 ±0.22	3.63 ±0.08	5.11 ±0.03	3.12 ±0.11	5.04 ±0.33
PCA 256 / +code	15.05 ±0.13	17.04 ±0.15	—	—	5.41 ±0.19	7.72 ±0.48
AE 256 / image	5.55 ±0.15	7.23 ±0.19	2.70 ±0.02	3.95 ±0.09	3.03 ±0.18	4.75 ±0.21
AE 256 / +code	4.90 ±0.11	6.42 ±0.15	—	—	2.75 ±0.17	4.62 ±0.14
VAE 256 / image	5.75 ±0.42	7.32 ±0.30	2.71 ±0.02	3.94 ±0.06	3.12 ±0.18	5.04 ±0.24
VAE 256 / +code	4.94 ±0.22	6.48 ±0.22	—	—	2.99 ±0.11	4.81 ±0.07
VAE 256 / +raw code (recipe)	4.83 ±0.14	6.28 ±0.25	—	—	2.99 ±0.21	4.75 ±0.13

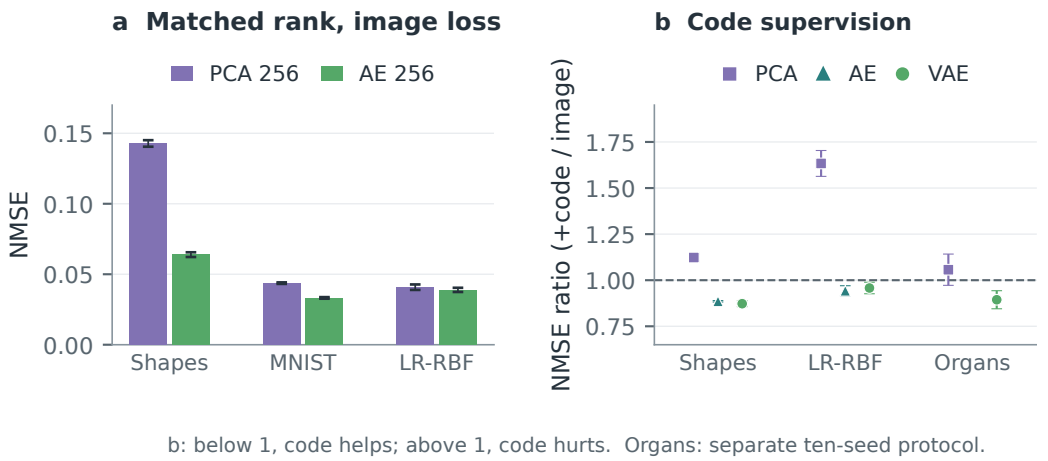

Figure 15: Matched-rank and code-supervision factorial, Transolver with 256 coordinates. (a) Image-only PCA and AE heads on the main test set, averaging noise 0 and 0.6 within each of three seeds. (b) NMSE with code loss divided by NMSE with image loss alone. Below one, code supervision helps. The synthetic classes use standardized code targets on the main test set. OrganAMNIST uses its separate ten-seed official-split protocol. Means and sample SDs over seeds. Tables 19 and 8 contain the values.

Same-code affine decoder. This post hoc check compares the reconstruction term U of Proposition 2 for two decoders fed the same codes. For each class and prior seed, we fit a dense affine decoder to the frozen encoder's posterior-mean codes of the training targets. We choose the ridge strength on a 20% holdout and refit on all training targets. We then decode the same codes of the 2,000 low-pass test targets with the affine map and with the nonlinear decoder. The affine decoder's error is 8 to 20 times as large on every class (Table 20). The encoder and nonlinear decoder were trained together, and the affine map was fitted afterward. So this shows how these codes decode, but it does not separate range nonlinearity from convolutional structure or from encoder–decoder coadaptation.

Table 20: Same-code decoding on the low-pass test set. Reconstruction NMSE of the frozen nonlinear decoder $\mathcal{D}$ and a fitted affine decoder L from identical encoder codes, averaged over three prior seeds. Intervals are paired-target bootstrap.

class	prior	nonlinear $\mathcal{D}$	affine L	$L/\mathcal{D} - 1$ (95% CI)
LR-RBF	AE	0.00024	0.0046	+1848% [+1772, +1933]
LR-RBF	VAE	0.00031	0.0046	+1376% [+1320, +1436]
MNIST	AE	0.0021	0.0417	+1862% [+1823, +1899]
MNIST	VAE	0.0025	0.0436	+1648% [+1614, +1683]
shapes	AE	0.0136	0.1089	+699% [+681, +716]
shapes	VAE	0.0131	0.1045	+698% [+683, +714]

Oracle code refinement. Proposition 1 uses an achieved reconstruction U_N, and Proposition 2 a reference code. The encoder roundtrip is one choice of each. In a post hoc check on 64 main-test-set Shapes (16 per family), we freeze the seed-42 VAE_{256} and optimize only its codes against the true images. We run 150 Adam steps at learning rate 0.03 from the encoder mean and two perturbed starts (SD 0.05), and keep the best iterate per object. Mean NMSE falls from 0.01257 for the roundtrip to 0.00380 (69.8% lower), against 0.03466 for the PCA_{256} projection of the same objects. NMSE uses the full main-test-set target variance (0.0577871). The roundtrip therefore overstates the distance to the decoder's range. Because the refined codes use the true targets as labels, the result is an achieved upper bound, not a certified floor.

Projecting spatial predictions. In an exploratory check on clean Shapes (low-pass test set), we fit decoder codes to the factorial's spatial predictions with 200 Adam steps, which approximately

projects them onto the frozen VAE's range. NMSE changes only from about 0.0630 to 0.0629, and ground truth is used only for scoring. So restricting a spatial prediction to the decoder's range after training does not reproduce the benefit of predicting coordinates directly. As Section E.3 shows, membership in a good range does not by itself make a predictor accurate.

Physical-response pilot. An exploratory pilot asks whether the heads respond differently to a small physical change. For eight clean pairs of translated circles and ellipses, we compare the change between the two reconstructions of each pair with the true displacement, using three saved seeds per image-only factorial head. Displacement NMSE is 0.181 for the spatial head, 0.617 for PCA_{256}, 0.163 for AE_{256}, and 0.180 for VAE_{256}. The nonlinear heads show no special advantage in local response, and PCA responds poorly. This is a check of local response, not evidence of an internal physical representation.

F Data and compute

This appendix gives the resource comparisons behind Section 6.5.

F.1 Paired data, clean targets, and model size

Table 21: **Complete Shapes resource sweeps (low-pass test set).** NMSE means and sample SDs over seeds 42, 123, and 456. VAE trainable counts exclude 2.28M frozen decoder parameters. Clean-target pretraining uses 100 epochs, so target count and optimizer steps covary.

Model / clean targets m	Trainable (M)	Epochs	Noise	NMSE	Seed SD
Spatial	0.329	120	0.0	0.08116	0.00434
Spatial	0.329	120	0.3	0.09979	0.00506
Spatial	0.668	120	0.0	0.06464	0.00209
Spatial	0.668	120	0.3	0.08152	0.00278
Spatial	1.305	120	0.0	0.06285	0.00335
Spatial	1.305	120	0.3	0.08745	0.00799
VAE anchor	1.472	120	0.0	0.04993	0.00102
VAE anchor	1.472	120	0.3	0.05428	0.00039
Spatial	2.655	120	0.0	0.06190	0.00271
Spatial	2.655	120	0.3	0.08071	0.00216
Spatial	5.193	120	0.0	0.06060	0.00085
Spatial	5.193	120	0.3	0.07880	0.00244
Spatial	10.583	120	0.0	0.05578	0.00353
Spatial	10.583	120	0.3	0.07041	0.00200
VAE anchor	1.472	300	0.0	0.04701	0.00093
Spatial	10.583	300	0.0	0.04682	0.00299
VAE / $m = 250$	1.472	120	0.0	0.21913	0.01928
VAE / $m = 250$	1.472	120	0.3	0.22022	0.01897
VAE / $m = 500$	1.472	120	0.0	0.10383	0.01137
VAE / $m = 500$	1.472	120	0.3	0.10628	0.01096
VAE / $m = 1000$	1.472	120	0.0	0.06761	0.00473
VAE / $m = 1000$	1.472	120	0.3	0.07105	0.00462
VAE / $m = 2000$	1.472	120	0.0	0.05467	0.00051
VAE / $m = 2000$	1.472	120	0.3	0.05918	0.00065

The sweeps use Transolver on Shapes and the low-pass test set, with clean measurements and, in the model-size and clean-target sweeps, noise 0.3 as well. They use different schedules: 3,000 updates for the paired-data sweep (Figure 16a), and 120 epochs for the model-size sweep, which pits six spatial models against the fixed VAE recipe (Figure 16b–c, Table 21). Extended to 300 epochs, the largest spatial model and the VAE recipe reach comparable clean error. This does not establish equivalence, and we do not extrapolate beyond the observed range.

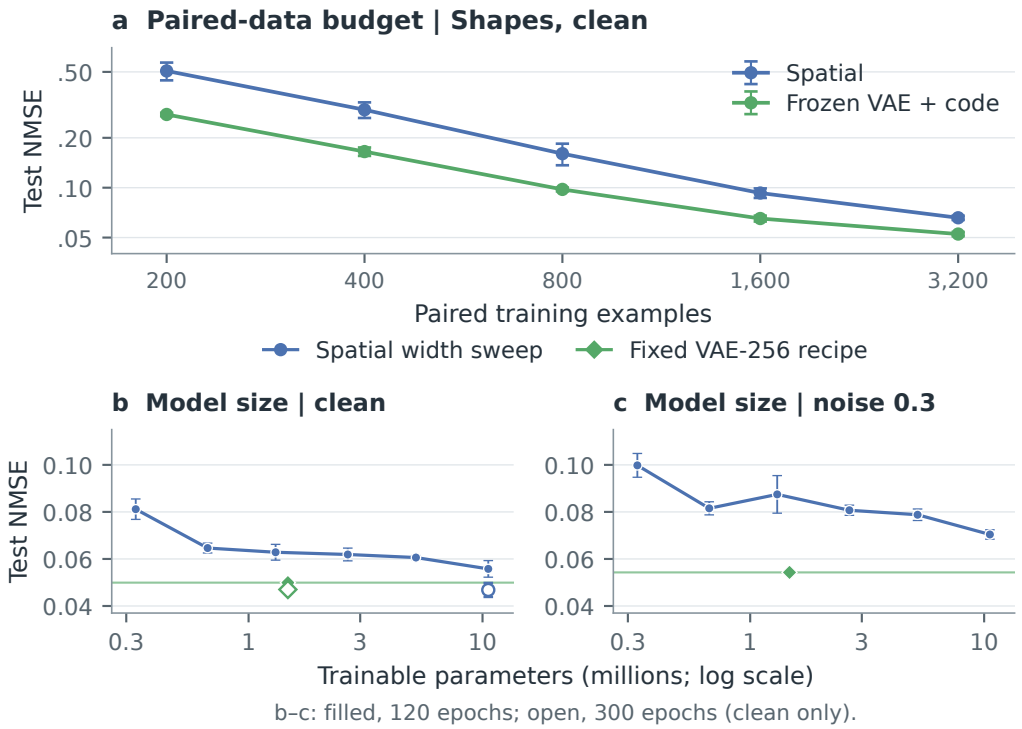

Figure 16: Paired data and model size on Shapes, low-pass test set. (a) Test NMSE against paired training examples after 3,000 updates. The pretrained head also uses 4,000 clean targets for its decoder. (b, c) Six spatial model sizes against the fixed VAE-256 recipe (1.47M trainable, plus the 2.28M frozen decoder), clean and at noise 0.3. Filled markers train for 120 epochs, open markers for 300. Schedules differ between (a) and (b, c). Means and sample SDs over three seeds. Table 21 contains the values for (b, c).

F.2 High-data benchmark

The high-data benchmark (Table 22) trains on 12,800 Shapes pairs, with 800 development pairs. It scores the Shapes main test set without the five objects that also occur in the enlarged training set (1,995 objects). The spatial model keeps each backbone's panel width and depth. The nonlinear model uses the convolutional coordinate head (Appendix A.2) and a frozen VAE-256 decoder fitted on the 12,800 training targets, trained with the loss of Eq. (4) on standardized codes. For the attention backbones, its coordinate head first passes the token grid through a stride-two 4×4 transposed convolution to width w and a 3×3 convolution, mirroring the spatial head's decoder, before downsampling. FFNO's feature map is already 64×64, so it skips this stage.

An exhaustive search over the nonlinear model's backbone width and head widths matches it to the spatial model within 5%, in trainable parameters and in batch-1 inference MACs including the frozen decoder (Table 23). Both heads train for 120 epochs with batch size 128, cosine decay, and paired minibatch order, over seeds 42, 123, and 456. For Galerkin, OFormer, GNOT, and FFNO, we choose each head's learning rate separately from $\{2.5, 5, 10\} \times 10^{-4}$ by development NMSE on seed 42. OFormer's spatial model fails to train at 10^{-3} (development NMSE 0.93). Transolver uses 10^{-3} for both heads, chosen in earlier tuning at this data size that we did not archive.

F.3 Compute per model

Table 24 separates trainable parameters, weights needed at inference, and arithmetic. In the main benchmark, the nonlinear models train 1–17% more parameters than the spatial ones and also store the 2.28M frozen decoder. On the four attention backbones, they need fewer MACs. In the matched controls, the pretrained model trains less than half the spatial model's parameters and takes about 60% as long per training step.

Table 22: **Nonlinear coordinates with four times more paired data.** Shapes, 12,800 training pairs. Heads are matched within 5% in trainable parameters and inference MACs, including the frozen decoder. NMSE mean and sample SD over three paired seeds on the test set (1,995 objects).

Backbone	Trainable (M)	Spatial	Nonlinear	Change (%)	Seed wins
Transolver	1.30 / 1.36	0.0313 $\pm$0.0008	**0.0283** $\pm$0.0001	-9.4	3/3
Galerkin	1.44 / 1.50	0.0445 $\pm$0.0014	**0.0320** $\pm$0.0003	-28.0	3/3
OFormer	1.43 / 1.47	0.0469 $\pm$0.0033	**0.0300** $\pm$0.0003	-36.1	3/3
GNOT	1.47 / 1.54	0.0417 $\pm$0.0061	**0.0311** $\pm$0.0002	-25.4	3/3
FFNO	1.70 / 1.72	0.0407 $\pm$0.0005	**0.0283** $\pm$0.0003	-30.4	3/3

Table 23: Per-backbone configurations in the high-data benchmark (Table 22). Backbone width and learning rate are given for the spatial and nonlinear models; w, w_2, k, and h are the convolutional coordinate-head widths (Appendix A). Ratios compare the nonlinear model with the spatial model; MACs are per sample and include the frozen decoder.

Backbone	Layers	Width	w	w_2	k	h	Params ratio	MACs ratio	Learning rate
Transolver	4	112 / 104	72	16	16	128	1.043	0.977	10^{-3} / 10^{-3}
Galerkin	5	168 / 168	72	16	8	128	1.045	0.962	5×10^{-4} / 10^{-3}
OFormer	5	152 / 136	112	32	8	128	1.034	0.974	5×10^{-4} / 5×10^{-4}
GNOT	5	144 / 144	56	24	8	128	1.044	0.956	10^{-3} / 10^{-3}
FFNO	4	80 / 72	16	16	16	256	1.011	0.978	10^{-3} / 10^{-3}

Table 24: **Compute per model.** Trainable: parameters updated in inverse training. Inference weights: trainable weights plus the frozen decoder (2.28M) or the PCA basis and mean (0.53M). The VAE encoder (4.29M), used only for code targets during training, is excluded. MACs: per reconstruction, counting matrix products and convolutions. For FFNO, complex products count as four real MACs and the FFTs (39M MACs) are estimated at $5n \log_2 n$ flops per complex length-n transform. The VAE recipes also run the encoder on every training target (43M MACs each). Matched controls: Transolver models of Table 2. Timing on one NVIDIA GeForce RTX 5090 (fp32, with TF32 off for matrix products and on for convolutions): batch-1 latency, batch-128 inference throughput, and batch-128 training step (medians, with VAE rows including the per-minibatch encoder pass).

Backbone	Head	Trainable	Inference weights	MACs	Latency (ms)	Throughput (s^{-1})	Train step (ms)	Peak mem. (GB)
Transolver	Spatial	1.30M	1.30M	1.45G	0.99	8,404	53.9	3.99
	PCA_{128}	1.44M	1.97M	1.23G	1.01	9,797	46.2	3.50
	VAE_{256}	1.47M	3.75M	1.26G	1.08	9,400	47.8	3.52
Galerkin	Spatial	1.44M	1.44M	1.73G	1.08	2,636	107.9	7.87
	PCA_{128}	1.41M	1.94M	1.24G	1.03	2,876	95.5	6.72
	VAE_{256}	1.44M	3.72M	1.26G	1.10	2,834	97.0	6.75
OFormer	Spatial	1.43M	1.43M	1.63G	1.28	2,193	120.2	7.87
	PCA_{128}	1.45M	1.98M	1.01G	1.03	3,052	87.2	6.03
	VAE_{256}	1.49M	3.76M	1.04G	1.10	3,007	88.5	6.06
GNOT	Spatial	1.47M	1.47M	1.70G	1.11	3,560	103.1	9.03
	PCA_{128}	1.53M	2.05M	1.34G	1.06	3,942	91.8	8.05
	VAE_{256}	1.56M	3.84M	1.36G	1.14	3,894	93.1	8.07
FFNO	Spatial	1.70M	1.70M	0.69G	0.60	4,170	77.0	4.81
	PCA_{128}	1.95M	2.48M	0.68G	0.63	4,232	76.2	4.82
	VAE_{256}	1.99M	4.26M	0.70G	0.72	4,143	77.7	4.84
Transolver (matched)	Spatial	3.75M	3.75M	3.87G	1.47	4,844	84.9	5.11
	Scratch	3.75M	3.75M	3.87G	1.38	4,451	93.2	7.43
	Pretrained	1.65M	3.93M	1.39G	1.14	8,596	50.8	3.70

With cached code targets, the VAE training step is about 1% faster. Fitting a VAE prior takes 8.7 ms per batch of 256, or 14 s of optimizer steps for 100 epochs on 4,000 targets. Inverse training runs 120 epochs: 3,000 (Shapes), 15,000 (LR-RBF), and 33,840 (MNIST) steps.