KAPI-ONET: KERNEL-ADAPTIVE PHYSICS-INFORMED OPERATOR NETWORK FOR PDES

Anonymous authors
Paper under double-blind review

ABSTRACT

Physics-informed operator learning enables rapid prediction across families of PDEs without requiring paired numerical solution fields, but existing methods often rely on latent or globally shared representations that are difficult to interpret physically and can be computationally expensive to train. This raises the question of whether an operator learner can expose how variations in the input reorganize the solution representation while remaining compact enough to reduce the computational burden of physics-informed training. To address this gap, we propose KAPI-ONet, a kernel-adaptive physics-informed operator network in which each solution is represented explicitly by a task-dependent collection of Gaussian RBFs with interpretable centers, scales, orientations, and coefficients in physical or space–time coordinates. For functional inputs, KAPI-ONet formulates operator learning as a mapping between structured Gaussian-splat representations of the input and solution, providing a common coordinate-aware description of both sides of the operator through geometrically meaningful amplitudes, locations, and scales. Across elliptic, time-dependent, and Navier–Stokes problems, the learned kernel geometry adapts to localized and high-gradient solution structures, while KAPI-ONet attains competitive accuracy with substantially lighter training in settings where direct computational comparisons are available. We also evaluate KAPI-ONet beyond standard forward PDE operator learning, including parameter-to-field prediction and two biomedical applications: inverse hemodynamics from sparse velocity measurements and weakly supervised cardiac annotation. These results support task-adaptive kernel geometry as an interpretable and computationally efficient representation for operator learning across diverse problem formulations. Code and reproducibility materials are available at: `https://anonymous.4open.science/status/kapi-onet-anonymous-6C97`

1 INTRODUCTION

Many problems in science and engineering, including iterative design, optimization, uncertainty quantification, inverse modeling, and real-time prediction or control, require solving closely related PDEs across a large number of problem instances. These instances may differ in their forcing functions, coefficients, initial or boundary conditions, or physical parameters. Conventional numerical solvers treat each instance independently, so the overall computational cost can become substantial when many repeated solutions are required. Operator learning offers an alternative by learning the map from a problem specification to its corresponding solution field, so that a single trained model can be evaluated on previously unseen instances. Neural operators extend standard neural networks from mappings between finite-dimensional vectors to mappings between function spaces, and have emerged as a promising framework for approximating solution operators of PDEs (Lu et al., 2021; Li et al., 2021). Physics-informed operator learning further reduces the dependence on precomputed solution fields by training the operator through the governing equations and associated boundary or initial conditions (Wang et al., 2021b).

Beyond how the operator is trained, an equally important question is how its output solution is represented. Existing approaches range from solution-basis methods, where basis functions directly construct the predicted field, to transform-basis methods, where Fourier, wavelet, or related representations parameterize internal operator transformations (Shin et al., 2024; Liu et al., 2025).

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.

DeepONet, for example, combines input-dependent coefficients with a globally learned trunk basis (Lu et al., 2021), while FNO and WNO perform nonlocal transformations in prescribed Fourier and wavelet coordinates (Li et al., 2021; Tripura & Chakraborty, 2023). These architectures have demonstrated strong predictive capability, but they also raise a fundamental representation question: should all members of a PDE family share essentially the same spatial representation, or should the basis itself adapt to the localized structures arising in each individual solution?

Despite substantial progress, existing operator-learning architectures still leave two related challenges unresolved. First, task dependence is often encoded through latent features, globally shared solution bases, or transform-domain representations, making it difficult to directly interpret how changes in the input reorganize the spatial or spatio-temporal structure of the predicted solution. This is particularly limiting when the solution contains localized, moving, or sharply varying features that would naturally benefit from a representation whose geometry adapts from one task to another. Second, physics-informed operator learners can require large models, extensive collocation evaluations, and long optimization schedules, creating a need for more compact representations that allocate capacity where it is most useful while retaining competitive predictive accuracy.

Localized radial-basis representations provide a natural route toward such a task-adaptive and interpretable operator representation. Classical RBF theory establishes their approximation power, including nonlinear operator approximation with input-dependent coefficients over a shared RBF geometry (Park & Sandberg, 1991; Chen & Chen, 1995), while physics-informed RBF methods show that adapting kernel locations and scales can act as an implicit form of mesh adaptivity (Ramabathiran & Ramachandran, 2021; Shao et al., 2026). Gaussian-splat PDE representations further demonstrate that localized anisotropic kernels can concentrate around physically important structures (Han et al., 2026), and Meta-SPINN shows that the RBF geometry itself can be conditioned on finite-dimensional PDE parameters (Dwivedi et al., 2026). These observations motivate KAPI-ONet: instead of using a single fixed solution basis, we learn an operator that generates both the geometry and coefficients of a localized kernel representation from the task, including the functional-input setting. Detailed connections to these methods are provided in Appendix A.

Contributions. Our main contributions are:

- We develop KAPI-ONet, an interpretable operator learner that generates a task-adaptive localized Gaussian-RBF representation of the solution, with centers, scales, orientations, and coefficients defined directly in physical or space–time coordinates.
- We formulate functional operator learning as a mapping between structured Gaussian-splat representations of the input and solution, providing a common coordinate-aware description of both sides of the operator through geometrically meaningful quantities such as amplitudes, locations, and scales.
- We demonstrate the generality of KAPI-ONet beyond standard forward PDE operator learning through parameter-to-field prediction and two biomedical applications, sparse-observation inverse hemodynamics and weakly supervised cardiac annotation, while achieving competitive accuracy with substantially lighter training in benchmark settings where direct computational comparisons are available.

Organization. The remainder of the paper is organized as follows. Section 2 presents KAPI-ONet, Section 3 reports the numerical and biomedical results, Section 4 discusses limitations and future work, and Section 5 concludes the paper. Appendices A–F provide detailed related work, baseline descriptions, mathematical formulation, test cases, results, and architecture details, while Appendices G and H report out-of-distribution evaluation and the parent–child factorization study.

2 Proposed Kernel-adaptive physics-informed operator network (KAPI-ONet)

KAPI-ONet represents each task by an explicit, task-dependent continuous kernel field rather than predicting the solution independently at every query coordinate. For a task T, a finite-dimensional representation $r(T)$ is mapped by a conditioning network to a kernel state that determines the geometry and expansion coefficients of the output field. The resulting continuous representation is

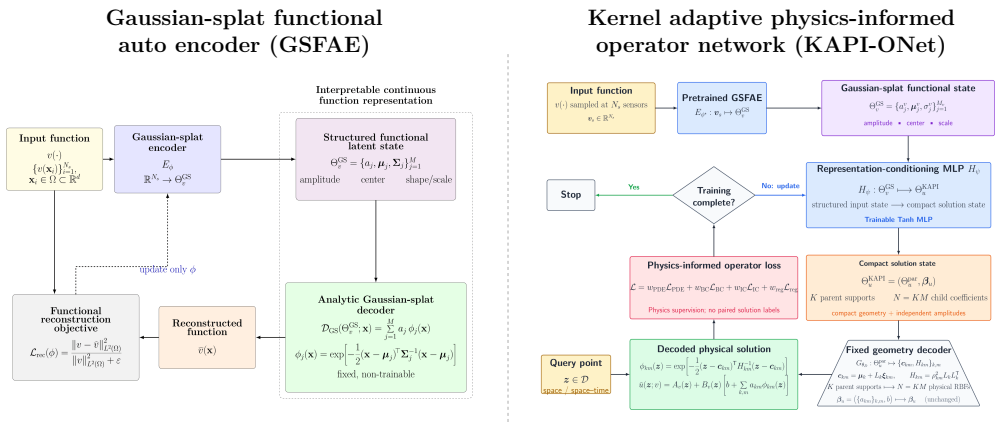

Figure 1: Overview of KAPI-ONet for functional-input PDE operator learning. Left: the Gaussian-splat functional autoencoder (GSFAE) maps sampled input functions to an interpretable continuous kernel representation using a trainable encoder and fixed analytic Gaussian decoder. Right: the frozen functional state is mapped by the representation-conditioning network to a compact task-dependent solution state. A fixed parent–child geometry decoder expands task-adaptive parent supports into physical Gaussian RBFs, whose independent coefficients define the continuous solution field. Physics-informed training updates the operator representation without paired solution labels. Full mathematical details are given in Appendix C.

generated once per task and can subsequently be evaluated at arbitrary spatial or space–time coordinates. The complete mathematical formulation is given in Appendix C.

Figure 1 summarizes the canonical realization used for functional-input PDEs. When the task is specified by an input function, we first construct a structured representation using a Gaussian-splat functional autoencoder (GSFAE). The trainable encoder maps discrete sensor observations to amplitudes, centers, and scales of Gaussian kernels, while a fixed analytic decoder reconstructs the continuous input function. Because the decoder is non-trainable, the resulting latent state is not an abstract feature vector but an explicit geometric description of the input. The GSFAE is pretrained using only function reconstruction and is then frozen during operator training.

The frozen functional state is passed to a representation-conditioning network, which generates a compact task-dependent solution state. Rather than predicting the geometry of every physical RBF independently, KAPI-ONet uses a hierarchical parent–child construction. A small set of anisotropic parent supports is adapted to the task through changes in position, scale, and orientation; a fixed geometry decoder then expands each parent into several physical child RBFs. The children inherit the parent transformation but retain independent expansion coefficients. This preserves a rich physical basis while substantially reducing the number of task-dependent geometric quantities that must be generated.

The decoded RBFs and their coefficients define an explicit continuous solution field. Query coordinates enter only when this field is evaluated, so the same task-conditioned state can be reused at arbitrary locations without regenerating the representation. When possible, boundary or initial conditions are embedded analytically into the trial field; the remaining unknowns are learned by minimizing PDE residuals and any additional constraint or regularization terms. Thus, operator training requires the task inputs and governing equations, but not paired numerical solution fields.

Although Figure 1 shows the functional-input PDE realization, the defining component of KAPI-ONet is the task-conditioned kernel representation of the output rather than the GSFAE itself. For low-dimensional parameter inputs, sparse observations, or images, the task representation can be supplied directly or produced by a problem-specific encoder, while the same general principle of generating an explicit task-dependent continuous field is retained. Detailed derivations, exact constraint constructions, parent–child parameterization, and training objectives are provided in Appendix C.

3 RESULTS AND DISCUSSION

All KAPI-ONet experiments were conducted entirely on CPU using a 13th Gen Intel(R) Core(TM) i7-13700 processor with 16 physical cores; no GPU acceleration was used.

3.1 FORWARD OPERATOR LEARNING

We first evaluate KAPI-ONet on forward physics-informed operator-learning problems spanning two-dimensional elliptic PDEs and one-dimensional time-dependent nonlinear PDEs. The five settings comprise two Poisson problems, diffusion–reaction, and two Burgers problems with parametric and Gaussian-random-field (GRF) initial conditions. Across all cases, the same computational principle is retained: the input function is encoded by the pretrained Gaussian-splat functional encoder, the structured task representation is mapped to task-dependent solution parents, and the fixed parent–child decoder generates the physical RBF dictionary used to construct the continuous solution field. Complete problem definitions are provided in Appendix D, detailed forward-operator results in Appendix E.1–E.3, and architecture details in Appendix F.

Figure 2 illustrates this common representation mechanism across all five benchmarks. In addition to the close agreement between the KAPI-ONet predictions and their numerical or analytical references, the task-conditioned parent supports provide a direct and interpretable view of how the solution representation adapts to each problem. For localized Poisson forcing, the parent supports concentrate around localized solution structures and contract where sharper spatial variation must be resolved. For the time-dependent Burgers problems, the anisotropic supports orient along evolving space–time structures and become narrower in regions of steep gradients, particularly for the GRF initial-condition case. The generated child RBFs inherit these task-dependent geometric transformations while retaining independent expansion coefficients. Thus, the same input-representation, task-conditioning, and parent–child decoding pipeline adapts its geometry across substantially different forcing and initial-condition families.

The quantitative benchmark comparison is summarized in Table 1. The table focuses on the four settings for which corresponding physics-informed neural-operator results are available; the localized-Gaussian Poisson case is primarily used to expose the adaptive representation mechanism and its complete results are reported in Appendix E.1.1. PI-WNO and PI-DeepONet attain lower absolute errors on their respective benchmark settings, while KAPI-ONet exhibits a favorable accuracy–training-cost trade-off. For diffusion–reaction, for example, KAPI-ONet attains a mean relative L_2 error of 0.76% with a complete training pipeline of less than 40 minutes on CPU, compared with 0.45% and approximately 40 hours for our same-platform PI-DeepONet reproduction. The PI-WNO comparisons exhibit a similar trend. For GRF Burgers, a hardware-normalized PI-DeepONet wall-clock comparison is not reported because a corresponding same-platform run is unavailable; its released implementation uses 200k optimization updates and notes possible out-of-memory issues on Google Colab, recommending server execution. Full error distributions, metric definitions, runtime methodology, and baseline details are given in Appendix E.1.2, Appendix E.2, and Appendix E.3.

Additional robustness and representation studies are reported in the Appendix. Appendix G evaluates out-of-distribution generalization under shifts in input statistics and physical parameters, while Appendix H shows that the parent–child factorization provides a controllable trade-off between task-adaptive geometric flexibility and parameter efficiency at a fixed physical RBF budget.

3.2 PARAMETRIC LEARNING

We next consider the lid-driven cavity (2D steady Navier–Stokes) as a finite-dimensional parameter-to-field problem, where the task is specified only by the Reynolds number $Re \in [10, 100]$. Unlike the preceding forward operator-learning problems, no functional input encoder is required; the normalized scalar parameter is supplied directly to the conditioner, which generates the task-dependent RBF representation of the velocity and pressure fields. No FEM solution fields are used during training or checkpoint selection. Complete problem, training, and architectural details are provided in Appendix D, Appendix E.4, and Appendix F.

Figure 3 compares KAPI-ONet with independent FEM references at representative low, intermediate, and high Reynolds numbers. The predicted velocity-magnitude fields reproduce the dominant

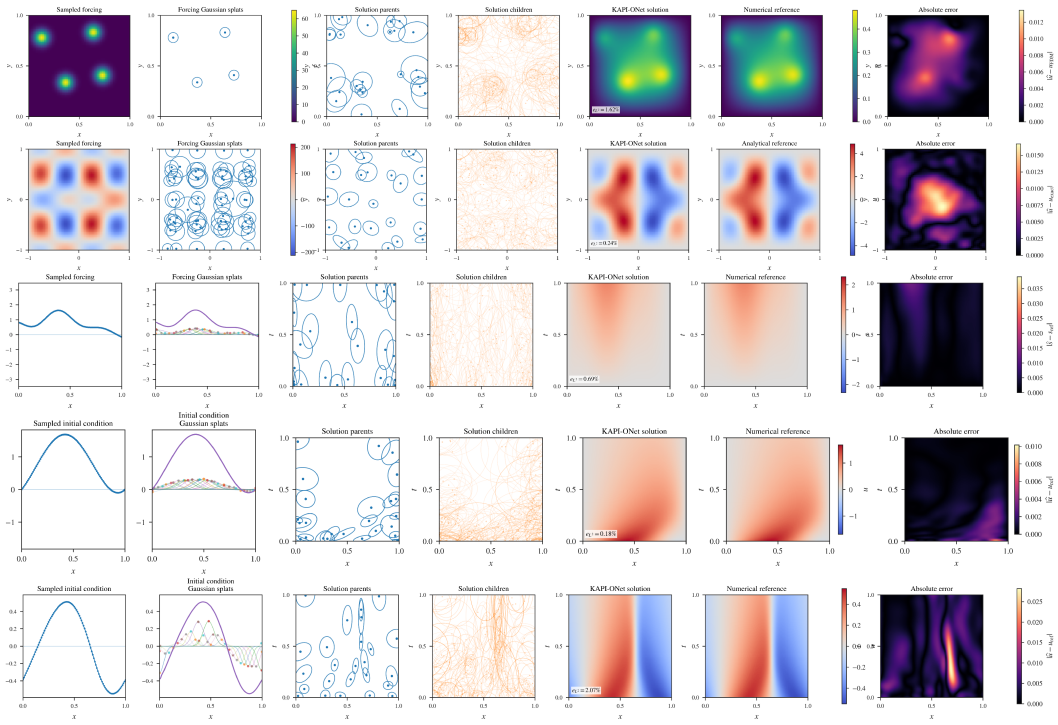

Figure 2: Representative forward operator-learning results for localized Poisson, sinusoidal Poisson, diffusion–reaction, parametric Burgers, and GRF Burgers (top to bottom). Columns show the sampled input function, its Gaussian-splat functional representation, the task-conditioned solution parents, generated child RBFs, KAPI-ONet prediction, numerical or analytical reference, and absolute error. The parent supports expose the task-adaptive geometry of KAPI-ONet: they concentrate and contract around localized structures in the Poisson problem and become anisotropic, oriented, and narrow around evolving high-gradient structures in the Burgers problems. The same GSFAE $\rightarrow$ task-conditioning $\rightarrow$ parent–child decoding pipeline is used across all five benchmarks.

lid-driven flow structure across the parameter range, with the largest discrepancies concentrated near the moving lid and upper recirculation region. The corresponding centerline profiles show particularly close agreement for the vertical-centerline horizontal velocity, while the horizontal-centerline vertical velocity becomes more sensitive toward larger Reynolds numbers.

Across seven independent FEM reference cases ($Re = \{10, 25, 40, 55, 70, 85, 100\}$), KAPI-ONet achieves mean relative L_2 errors of 15.55% for velocity magnitude and 20.97% for the complete velocity vector. The classical centerline diagnostics yield mean errors of approximately 9.75% for $u(0.5, y)$ and 16.15% for $v(x, 0.5)$. The complete physics-informed training requires approximately 2.23 h on CPU. This experiment therefore demonstrates that the defining KAPI-ONet mechanism is the generation of a task-conditioned continuous kernel representation of the output, rather than any particular form of input representation: the same framework naturally extends from function-to-field operators to parameter-to-field solution families.

3.3 Biomedical Applications

3.3.1 Inverse Operator Learning: Hemodynamics

We next consider a sparse-observation inverse hemodynamics problem, where the task is specified by velocity measurements at only five interior sensor locations. From these ten scalar measurements, KAPI-ONet reconstructs the continuous velocity and pressure fields and the corresponding wall shear stress (WSS), while the task-specific inlet profile remains hidden during inference. Full FEM fields are used only for post-training evaluation. Complete problem and training details are provided in Appendix D, Appendix E.5, and Appendix F.

Table 1: Forward physics-informed operator-learning benchmarks. KAPI-ONet is trained without paired solution fields. KAPI-ONet errors are mean relative L_2 errors. PI-WNO values marked with † are the relative-MSE values reported using the PI-WNO evaluation metric, whereas PI-DeepONet values are mean relative L_2 errors. KAPI-ONet training times include the complete measured GS-FAE and operator-training pipeline. PI-WNO times marked with ‡ are projected from same-platform CPU reproductions using the reported training regime; the diffusion–reaction PI-DeepONet time is from a same-platform CPU reproduction. For GRF Burgers, no hardware-normalized PI-DeepONet wall-clock time is reported. Complete benchmarking details are provided in Appendix E.1.2, Appendix E.2, and Appendix E.3.

PROBLEM	KAPI-ONET ERROR (%)	BASELINE	BASELINE ERROR (%)	TRAINING COST KAPI / BASELINE
Poisson (sinusoidal)	0.39	PI-WNO (N. et al., 2024)	0.0469†	45.8 min / ∼25 h‡
Diffusion–reaction	0.76	PI-DeepONet (Wang et al., 2021b)	0.45	<40 min / ∼40 h
Burgers (parametric IC)	0.28	PI-WNO	0.0518†	36.5 min / 3.9–5.2 h‡ 200k updates;
Burgers (GRF IC)	3.51	PI-DeepONet	2.80 / 1.38§	1.83 h / OOM warning

† PI-WNO metric; the corresponding KAPI-ONet values evaluated with the released PI-WNO metric are reported in the appendix. ‡ Same-platform CPU estimate based on the measured per-epoch runtime. § PI-DeepONet results for the loss-weight-tuned standard MLP and the modified, loss-weight-tuned architecture, respectively.

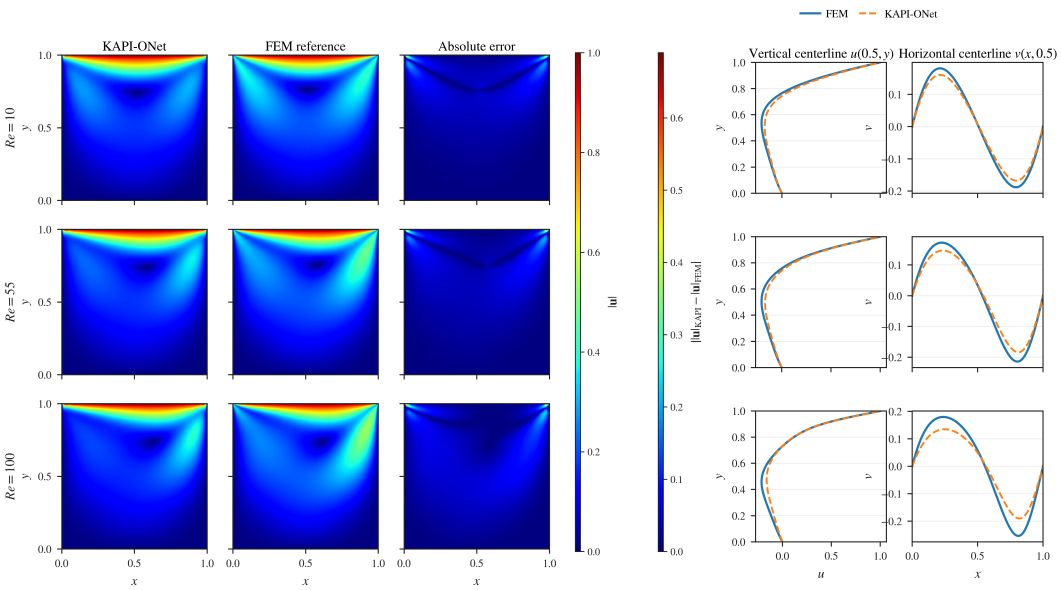

Figure 3: Parametric lid-driven cavity results at $Re = 10$, 55, and 100. Left: KAPI-ONet velocity magnitude, independent FEM reference, and pointwise absolute error. Right: classical cavity centerline diagnostics, comparing the vertical-centerline horizontal velocity $u(0.5, y)$ and the horizontal-centerline vertical velocity $v(x, 0.5)$ against FEM. KAPI-ONet is conditioned only on the Reynolds number and is trained without FEM solution-field supervision.

Figure 4 illustrates a representative held-out case. The predicted velocity field reproduces the dominant downstream flow structure despite a substantial discrepancy in the inferred inlet profile. This behavior reflects the underdetermined nature of the inverse problem: the five interior velocity measurements do not uniquely identify the hidden upstream boundary condition, and different inlet profiles can generate similar downstream flow behavior. Importantly, this ambiguity does not prevent accurate recovery of the clinically relevant quantity of interest. Across 50 held-out cases, the mean relative L_2 errors are 21.51% for velocity and 27.60% for pressure, while the hidden inlet profile has

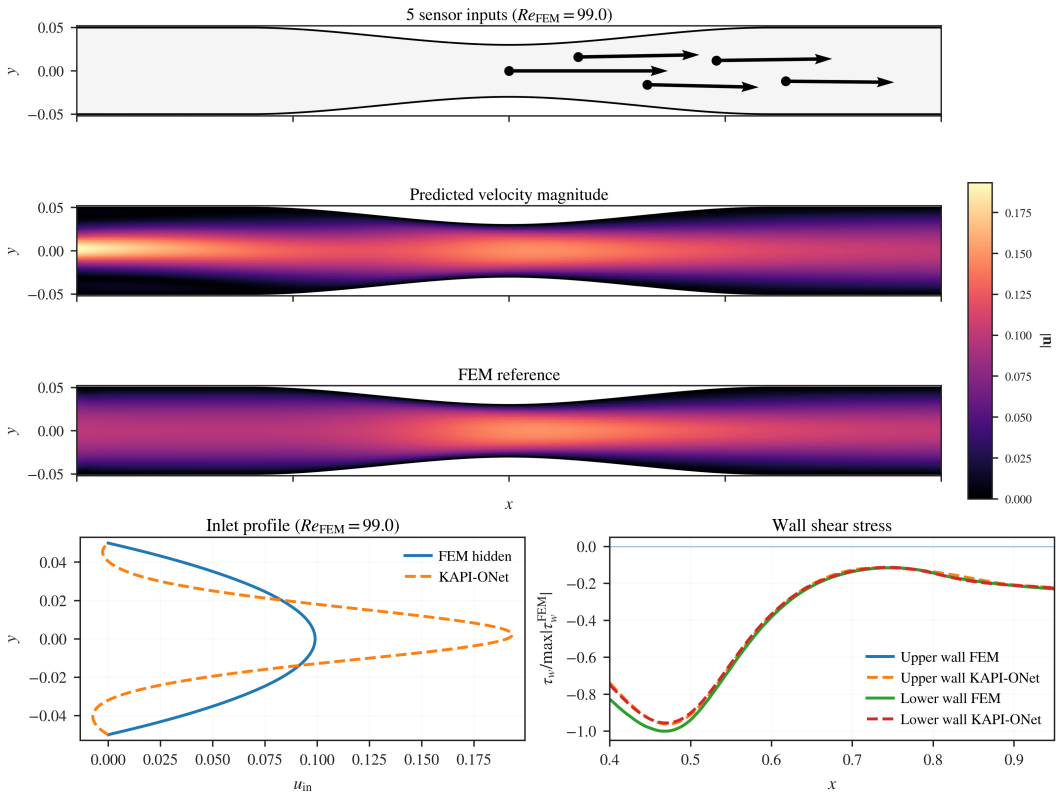

Figure 4: Representative sparse-observation inverse hemodynamics result. The task is specified only by velocity measurements at five interior sensors (top). KAPI-ONet reconstructs the continuous velocity field, shown together with the independent FEM reference. Although the inferred inlet profile differs substantially from the hidden FEM inlet condition, the downstream wall shear stress closely follows the FEM reference on both vessel walls. This illustrates the non-uniqueness of the hidden boundary-condition recovery while preserving accurate prediction of the downstream quantity of interest.

a much larger mean error of 84.43%. Nevertheless, the corresponding flow rate is recovered with only 4.30% mean error, and the combined upper- and lower-wall WSS achieves a mean relative L_2 error of 4.51% and a median error of 3.02%.

This result is consistent with the observations of Arzani et al. (Arzani et al., 2021), who showed that sparse interior velocity measurements can support accurate near-wall flow and WSS inference even when inlet and outlet boundary conditions are not fully known. In the present setting, KAPI-ONet extends this idea from an instance-specific inverse solve to an amortized inverse operator over a family of stenotic flows.

3.3.2 Variational operator learning for cardiac annotation

Our final application considers weakly supervised annotation of the left-ventricular (LV) cavity in cardiac MR images. This setting differs fundamentally from conventional supervised segmentation: dense expert masks are not supplied to KAPI-ONet during training. Instead, each task consists of an MR image and a single clinician-provided point inside the LV cavity, and the model generates a continuous implicit RBF field whose zero level set defines the predicted contour. Training is driven by an image-based variational objective based principally on the Chan–Vese region-fitting energy, together with weak localization and geometric regularization. The complete problem formulation is given in Appendix D, detailed results and the variational correction procedure in Appendix E.6, and architecture details in Appendix F.

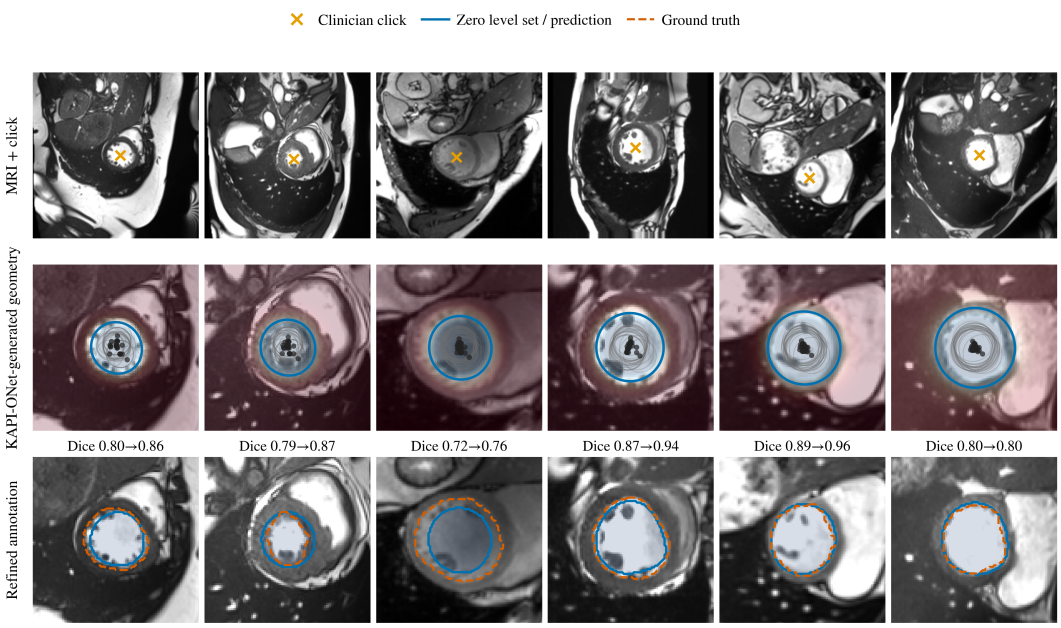

Figure 5: Weakly supervised variational operator learning for cardiac annotation. Top: cardiac MR images with the single clinician-provided localization point. Middle: KAPI-ONet-generated task-dependent RBF geometry and the corresponding zero-level-set prediction. Bottom: annotation after the fixed mask-free variational correction, shown retrospectively against the expert contour. KAPI-ONet is trained without dense expert masks; the reported expert contours are used only for validation and held-out evaluation. The operator provides an immediate image-specific annotation, which can optionally be refined through a short per-image variational correction.

The purpose of this experiment is therefore not to compete directly with fully supervised segmentation architectures such as U-Net, which learn from dense expert masks. Rather, we ask whether the variational segmentation principle, conventionally applied through optimization of each image individually, can be amortized across an image population as an operator-learning problem. KAPI-ONet learns this mapping once and subsequently produces an immediate task-specific contour representation for an unseen image; the generated RBF state may then optionally be refined by a short mask-free variational correction.

Figure 5 shows representative held-out cases. Starting from only the MR image and the clinician click, KAPI-ONet generates an interpretable image-specific RBF geometry and an initial zero-level-set annotation. The optional variational correction adapts this generated geometry to the individual image while remaining anchored to the operator prediction. Across 75 test images from 15 independently held-out subjects, the frozen operator achieves a mean Dice score of 0.868 ± 0.068. The fixed mask-free correction increases this to 0.912 ± 0.062 and improves 82.7% of the held-out cases. The frozen operator requires approximately 0.041 s per image on CPU, while the optional correction requires approximately 4.2 s per image. Expert masks are used only for validation-based hyperparameter selection and retrospective test evaluation, not as supervision for either the operator prediction or the per-image correction.

This experiment illustrates a broader use of KAPI-ONet beyond differential equations: the task-dependent kernel representation can amortize a family of variational optimization problems into a reusable operator, providing a rapid initial annotation that can subsequently be refined without dense mask supervision.

4 Limitations and Future Work

The first limitation concerns the degree of parameter sharing in the parent–child kernel factorization. Sharing child geometry through a smaller number of task-adaptive parents reduces the number of

task-dependent parameters and therefore the size of the conditioner, but it also restricts the independent degrees of freedom available to the physical RBFs. Appendix H shows that this creates a trade-off between parameter efficiency and geometric flexibility: stronger sharing reduces parameterization, whereas increasing the number of independently adaptive parents can improve accuracy. At present, however, there is no principled rule for selecting the appropriate amount of sharing for a given problem without unnecessarily increasing model size or degrading predictive accuracy.

A second limitation concerns robustness outside the training distribution. The out-of-distribution experiments in Appendix G show that performance degrades as the input statistics or physical parameters move beyond the training range. In particular, for GRF-driven problems, reducing the correlation length introduces sharper spatial variations than those seen during training, which leads to larger reconstruction and operator errors. This suggests that part of the OOD sensitivity can originate from the frozen GSFAE representation itself when its fixed kernel budget is insufficient to resolve previously unseen small-scale structures. Future work could therefore consider joint adaptation of the input representation and operator, together with mechanisms that increase kernel resolution when sharper features are encountered.

Finally, the present study focuses primarily on relatively low-dimensional spatial and space–time problems and uses a fixed family of Gaussian kernels with a predetermined parent–child decoding structure. Extending KAPI-ONet to three-dimensional PDEs, complex geometries, and multiphysics systems may require more flexible kernel families, geometry-aware parameterizations, and adaptive strategies for adding, removing, or refining basis functions across tasks. The biomedical experiments should likewise be viewed as proof-of-concept demonstrations; larger clinical datasets, realistic measurement noise, and broader validation will be required to assess their practical utility.

5 Conclusion

This work introduced KAPI-ONet, a physics-informed operator-learning framework in which each task generates an explicit continuous Gaussian-RBF solution representation with interpretable coefficients, locations, scales, and orientations. For functional inputs, the corresponding Gaussian-splat representation of the input provides a common coordinate-aware description on both sides of the operator, allowing changes in the input geometry to be related directly to changes in the solution representation. Across the considered PDE families, the learned kernels reorganize around localized structures, steep gradients, and evolving space–time features, while the parent–child factorization provides a compact mechanism for generating a richer physical RBF dictionary.

The results expose a clear trade-off between parameter sharing, geometric flexibility, and predictive accuracy: stronger parent–child sharing reduces the task-dependent parameterization and training burden, whereas greater independent adaptation provides additional representational freedom. Across forward PDE operators, parameter-to-field learning, inverse hemodynamics from sparse velocity measurements, and weakly supervised cardiac annotation, KAPI-ONet demonstrates that task-conditioned kernel geometry can provide a common interpretable representation across substantially different operator-learning settings. Future work will focus on principled selection of the degree of kernel sharing, adaptive kernel resolution for sharp and out-of-distribution structures, and extensions to higher-dimensional PDEs, complex geometries, and larger biomedical datasets.

References

Amirhossein Arzani, Jian-Xun Wang, and Roshan M. D'Souza. Uncovering near-wall blood flow from sparse data with physics-informed neural networks. *Physics of Fluids*, 33(7):071905, 07 2021. ISSN 1070-6631. doi: 10.1063/5.0055600. URL https://doi.org/10.1063/5.0055600.

Reza Azad, Ehsan Khodapanah Aghdam, Amelie Rauland, Yiwei Jia, Atlas Haddadi Avval, Afshin Bozorgpour, Sanaz Karimijafarbigloo, Joseph Paul Cohen, Ehsan Adeli, and Dorit Merhof. Medical image segmentation review: The success of u-net. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10076–10095, 2024. doi: 10.1109/TPAMI.2024.3435571.

Olivier Bernard, Alain Lalande, Clement Zotti, Frederick Cervenansky, Xin Yang, Pheng-Ann Heng, Irem Cetin, Karim Lekadir, Oscar Camara, Miguel Angel Gonzalez Ballester, Gerard San-

roma, Sandy Napel, Steffen Petersen, Georgios Tziritas, Elias Grinias, Mahendra Khened, Varghese Alex Kollerathu, Ganapathy Krishnamurthi, Marc-Michel Rohé, Xavier Pennec, Maxime Sermesant, Fabian Isensee, Paul Jäger, Klaus H. Maier-Hein, Peter M. Full, Ivo Wolf, Sandy Engelhardt, Christian F. Baumgartner, Lisa M. Koch, Jelmer M. Wolterink, Ivana Išgum, Yeonggul Jang, Yoonmi Hong, Jay Patravali, Shubham Jain, Olivier Humbert, and Pierre-Marc Jodoin. Deep learning techniques for automatic mri cardiac multi-structures segmentation and diagnosis: Is the problem solved? *IEEE Transactions on Medical Imaging*, 37(11):2514–2525, 2018. doi: 10.1109/TMI.2018.2837502. URL https://doi.org/10.1109/TMI.2018.2837502.

T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001. doi: 10.1109/83.902291. URL https://doi.org/10.1109/83.902291.

Tianping Chen and Hong Chen. Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks. *IEEE Transactions on Neural Networks*, 6(4):904–910, 1995. doi: 10.1109/72.392252.

Junho Choi, Taehyun Yun, Namjung Kim, and Youngjoon Hong. Spectral operator learning for parametric pdes without data reliance. *Computer Methods in Applied Mechanics and Engineering*, 420:116678, 2024. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2023.116678. URL https://www.sciencedirect.com/science/article/pii/S0045782523008010.

Vikas Dwivedi, Balaji Srinivasan, and Ganapathy Krishnamurthi. Physics informed contour selection for rapid image segmentation. *Scientific Reports*, 14(1):6996, Mar 2024. ISSN 2045-2322. doi: 10.1038/s41598-024-57281-x. URL https://doi.org/10.1038/s41598-024-57281-x.

Vikas Dwivedi, Monica Sigovan, and Bruno Sixou. Meta-spinn: meta-learned basis adaptation for parametric pdes. *Machine Learning: Science and Technology*, 7(4):045060, aug 2026. doi: 10.1088/2632-2153/ae94e2. URL https://doi.org/10.1088/2632-2153/ae94e2.

V. Fanaskov and I. Oseledets. Spectral neural operators, 2024. URL https://arxiv.org/abs/2205.10573.

Somdatta Goswami, Aniruddha Bora, Yue Yu, and George Em Karniadakis. Physics-informed deep neural operator networks, 2022. URL https://arxiv.org/abs/2207.05748.

Xu Han, Kuangxu Chen, Rui Wang, Guanzhou Wei, Ligong Han, and Han Gao. Toward nonlinear representations with gaussian-splat manifolds for physics-informed learning. *Nature Communications*, 17(1):9708, Aug 2026. ISSN 2041-1723. doi: 10.1038/s41467-026-76254-4. URL https://doi.org/10.1038/s41467-026-76254-4.

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

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

Shengjun Liu, Yu Yu, Ting Zhang, Hanchao Liu, Xinru Liu, and Deyu Meng. Architectures, variants, and performance of neural operators: A comparative review. *Neurocomputing*, 648:130518, 2025. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2025.130518. URL https://www.sciencedirect.com/science/article/pii/S0925231225011907.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, Mar 2021. ISSN 2522-5839. URL https://doi.org/10.1038/s42256-021-00302-5.

Lu Lu, Xuhui Meng, Shengze Cai, Zhiping Mao, Somdatta Goswami, Zhongqiang Zhang, and George Em Karniadakis. A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data. *Computer Methods in Applied Mechanics and Engineering*, 393:114778, 2022. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2022.114778. URL https://www.sciencedirect.com/science/article/pii/S0045782522001207.

Navaneeth N., Tapas Tripura, and Souvik Chakraborty. Physics informed wno. *Computer Methods in Applied Mechanics and Engineering*, 418:116546, 2024. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2023.116546. URL https://www.sciencedirect.com/science/article/pii/S0045782523006709.

Nicholas H. Nelsen and Andrew M. Stuart. Operator learning using random features: A tool for scientific computing. *SIAM Review*, 66(3):535–571, 2024. doi: 10.1137/24M1648703. URL https://doi.org/10.1137/24M1648703.

J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 06 1991. ISSN 0899-7667. doi: 10.1162/neco.1991.3.2.246. URL https://doi.org/10.1162/neco.1991.3.2.246.

Amuthan A. Ramabathiran and Prabhu Ramachandran. Spinn: Sparse, physics-based, and partially interpretable neural networks for pdes. *Journal of Computational Physics*, 445:110600, 2021. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2021.110600. URL https://www.sciencedirect.com/science/article/pii/S0021999121004952.

Zihan Shao, Konstantin Pieper, and Xiaochuan Tian. Solving nonlinear pdes with sparse radial basis function networks. *Journal of Machine Learning Research*, 27(177):1–58, 2026. URL http://jmlr.org/papers/v27/25-1399.html.

Xue Shi and Chunming Li. Convexity preserving level set for left ventricle segmentation. *Magnetic Resonance Imaging*, 78:109–118, 2021. ISSN 0730-725X. doi: https://doi.org/10.1016/j.mri.2021.02.003. URL https://www.sciencedirect.com/science/article/pii/S0730725X21000151.

Yeonjong Shin, Zhongqiang Zhang, and George Em Karniadakis. Chapter 6 - theoretical foundations of physics-informed neural networks and deep neural operators: A brief review. In Siddhartha Mishra and Alex Townsend (eds.), *Numerical Analysis Meets Machine Learning*, volume 25 of *Handbook of Numerical Analysis*, pp. 293–358. Elsevier, 2024. doi: https://doi.org/10.1016/bs.hna.2024.05.008. URL https://www.sciencedirect.com/science/article/pii/S1570865924000085.

Tapas Tripura and Souvik Chakraborty. Wavelet neural operator for solving parametric partial differential equations in computational mechanics problems. *Computer Methods in Applied Mechanics and Engineering*, 404:115783, 2023. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2022.115783. URL https://www.sciencedirect.com/science/article/pii/S0045782522007393.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021a. doi: 10.1137/20M1318043. URL https://doi.org/10.1137/20M1318043.

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

Appendix

KAPI-ONet: Kernel-Adaptive Physics-Informed Operator Network for PDEs

Anonymous submission

CONTENTS

Appendix A: Detailed related work 13

Appendix B: Physics-informed operator baselines 14

Appendix C: Detailed mathematical formulation of KAPI-ONet 15

Appendix D: Test cases 21

Appendix E: Detailed results 26

Appendix F: Architecture details 38

Appendix G: Out-of-distribution performance evaluation 41

Appendix H: Effect of parent–child factorization at fixed RBF budget 42

A DETAILED RELATED WORK

Operator learning for PDEs seeks to approximate the solution map associated with a family of differential equations rather than solving each problem instance independently. Let $v \in \mathcal{V}$ denote the input function specifying a problem instance, such as a forcing function, coefficient field, or initial condition, and let $u \in \mathcal{U}$ denote the corresponding solution field. The PDE family therefore induces a solution operator

$$\mathcal{G} : \mathcal{V} \longrightarrow \mathcal{U}, \qquad v \longmapsto u, \tag{A.1}$$

and a neural operator constructs a parameterized approximation $\mathcal{G}_\theta$ that generalizes this mapping to previously unseen inputs. In data-driven operator learning, $\mathcal{G}_\theta$ is trained from paired input–solution examples, whereas physics-informed operator learning constrains the predicted solution through the governing equations and associated boundary or initial conditions. Existing neural-operator architectures differ substantially in how the output field is represented and how basis functions enter the operator approximation (Shin et al., 2024; Liu et al., 2025).

From the perspective of basis representation, we organize these approaches into two broad classes: *solution-basis neural operators*, in which basis functions directly construct the predicted solution field, and *transform-basis neural operators*, in which a prescribed basis is used primarily to parameterize transformations within the operator layers.

Basis representations in neural operators. Solution-basis neural operators can be written in the general form

$$\widehat{u}(\mathbf{z}; v) = b(v) + \sum_{j=1}^{N} a_j(v)\phi_j(\mathbf{z}; v), \tag{A.2}$$

where $\mathbf{z}$ is an output coordinate, $a_j(v)$ are expansion coefficients, and ϕ_j are the functions spanning the output representation. For a given realization of the basis, Eq. A.2 is linear in the expansion coefficients, while the construction of the basis itself may range from fully prescribed to input dependent.

Analytic fixed bases are selected a priori from known function families. Spectral Neural Operators employ prescribed Fourier or Chebyshev expansions (Fanaskov & Oseledets, 2024), while SCLON predicts coefficients of orthogonal spectral expansions such as Fourier and Legendre bases (Choi et al., 2024). Data-derived fixed bases instead use information from the training solution ensemble; POD-DeepONet, for example, replaces the DeepONet trunk by proper orthogonal decomposition modes extracted from solution snapshots (Lu et al., 2022). DeepONet learns the output basis jointly with the coefficient map: its trunk functions form a globally learned basis that is shared across input instances, while its branch network produces the corresponding input-dependent coefficients (Lu et al., 2021). Physics-informed DeepONets retain this representation while replacing or augmenting paired-data supervision with constraints derived from the governing equations (Wang et al., 2021b).

Task-adaptive constructions allow the effective basis itself to depend on the input. Shift-DeepONet introduces input-dependent transformations of the learned trunk functions and can therefore be interpreted as producing an input-dependent effective output basis (Lanthaler et al., 2023). Function-valued random-feature models provide another example: although their random feature parameters are fixed after sampling, evaluating the function-valued features at a particular input produces output functions that depend on that input (Nelsen & Stuart, 2024). Thus, within solution-basis methods, one may distinguish analytic fixed, data-derived fixed, globally learned, and task-adaptive bases.

Transform-basis neural operators use basis functions differently. FNO parameterizes nonlocal operator transformations through Fourier coefficients and learned spectral multipliers (Li et al., 2021), while WNO performs the corresponding operation in a wavelet representation (Tripura & Chakraborty, 2023). Because these transform-domain operations are composed with pointwise maps and nonlinear activations, the final solution is generally not a single finite linear expansion in the Fourier or wavelet basis. The prescribed basis instead provides the coordinate system in which the internal nonlocal operator action is parameterized.

Localized and radial solution bases. Of particular relevance to KAPI-ONet is the long line of work on localized radial basis representations. Park and Sandberg established universal approximation for single-hidden-layer RBF networks (Park & Sandberg, 1991). Chen and Chen subsequently

extended RBF approximation to nonlinear functionals and operators (Chen & Chen, 1995). Their construction is especially relevant from the solution-basis perspective: the centers and scales of the output RBFs can be selected independently of the input function, while the expansion coefficients are continuous functionals of that input. Consequently, continuous nonlinear operators can be approximated using task-dependent coefficients over a common RBF output geometry.

This classical result establishes the expressive sufficiency of a shared RBF basis, but does not address whether adapting the basis geometry to an individual solution can provide a more efficient finite representation. A complementary line of physics-informed PDE solvers provides substantial motivation in this direction. SPINN reinterprets meshless RBF approximation as a shallow physics-informed neural network in which the expansion coefficients, kernel centers, and kernel widths are trainable, yielding an implicit form of mesh adaptivity (Ramabathiran & Ramachandran, 2021). More recently, Shao, Pieper, and Tian developed a sparse RBF formulation in which Gaussian features are adaptively inserted, optimized, and pruned while their centers and scales are treated as trainable parameters (Shao et al., 2026). Their analysis places the representation in a reproducing kernel Banach space, establishes finite sparse representations through a representer theorem, and provides convergence and error results for the resulting PDE approximation.

Related evidence is provided by Gaussian-splat representations for PDEs, where the solution is represented by mobile anisotropic Gaussian kernels whose locations, scales, and orientations are optimized or evolved according to the governing equations (Han et al., 2026). Such representations have been shown to concentrate geometric degrees of freedom around localized solution structures, including discontinuities and thin shear layers. These works operate primarily in the single-instance setting: when the PDE instance changes, the localized representation is optimized or evolved again. They nevertheless provide both analytical and computational motivation for allowing the geometry of a localized solution basis to respond to the solution being represented.

Meta-SPINN takes the next step in the parametric PDE setting by mapping task parameters to a shallow RBF representation with task-dependent coefficients, centers, scales, and directional parameters (Dwivedi et al., 2026). The resulting basis geometry therefore changes across the parametric family rather than being optimized independently for each new problem or shared globally by all tasks. Numerical experiments show that the learned localized bases reorganize according to the physical structures arising across the family, providing direct evidence that task-conditioned basis geometry is a viable representation for parametric PDE solution maps.

Taken together, these results provide a natural motivation for the task-conditioned kernel representation considered in KAPI-ONet. Classical RBF theory establishes universal approximation at the function level, while Chen and Chen show that nonlinear operators can already be approximated using task-dependent coefficients over shared RBF geometry. Physics-informed RBF and Gaussian-splat solvers provide complementary evidence that adapting localized basis geometry to an individual solution can yield compact and interpretable representations. Meta-SPINN demonstrates that the basis geometry itself can be conditioned on finite-dimensional PDE task parameters. KAPI-ONet builds on this progression by representing the solution through kernels whose expansion coefficients and geometry are generated from the task representation, including settings in which the task is specified by functional inputs. Moreover, provided that the task-conditioning map can recover task-independent kernel parameters and the corresponding input-dependent coefficients, the task-conditioned representation contains the fixed-basis RBF operator class as a special case and therefore inherits its universal approximation guarantee under the assumptions of Chen & Chen (1995).

B PHYSICS-INFORMED OPERATOR BASELINES

We consider two physics-informed neural operators as baselines: physics-informed DeepONet (PI-DeepONet) (Wang et al., 2021b) and the physics-informed wavelet neural operator (PI-WNO) (N. et al., 2024). These methods were selected as representative architectures from the two basis categories discussed in Section A: PI-DeepONet is a *solution-basis neural operator*, whereas PI-WNO is a *transform-basis neural operator*. This provides two structurally different reference models under a common physics-informed training setting.

Both methods approximate a parametric solution operator

$$\mathcal{G} : a \longmapsto u, \tag{B.1}$$

without requiring paired numerical solution fields. Instead, training is performed using the governing PDE together with the prescribed boundary and initial conditions. A generic objective takes the form

$$\mathcal{L}_{\text{phys}} = \lambda_r \mathcal{L}_r + \lambda_b \mathcal{L}_b + \lambda_i \mathcal{L}_i, \tag{B.2}$$

where the individual terms penalize the PDE residual and the applicable boundary and initial conditions. The principal differences between the two baselines concern how the solution operator is represented and how the derivatives entering the physics loss are evaluated.

Physics-informed DeepONet. PI-DeepONet retains the branch–trunk representation of DeepONet (Lu et al., 2021),

$$\mathcal{G}_\theta(a)(\mathbf{y}) = \sum_{k=1}^{q} b_k(\mathbf{a})\, t_k(\mathbf{y}), \tag{B.3}$$

where $\mathbf{a} = [a(x_1), \ldots, a(x_m)]$ denotes sampled values of the input function. The branch network produces input-dependent expansion coefficients, while the trunk network provides the globally learned solution basis. Because the output coordinates $\mathbf{y}$ enter the representation explicitly, the derivatives required by the governing equations can be evaluated through automatic differentiation. PI-DeepONet is used as a reference for the Burgers problem with GRF initial conditions and for the nonlinear diffusion–reaction problem.

Physics-informed wavelet neural operator. PI-WNO extends WNO (Tripura & Chakraborty, 2023) to physics-informed operator learning (N. et al., 2024). Rather than explicitly expanding the solution in a learned output basis, WNO parameterizes nonlocal transformations of discretized latent fields in a wavelet representation. A representative wavelet layer can be written as

$$v_{\ell+1} = \sigma\left(W_\ell v_\ell + \mathcal{W}^{-1}\left[R_\ell \mathcal{W}(v_\ell)\right]\right), \tag{B.4}$$

where $\mathcal{W}$ and $\mathcal{W}^{-1}$ denote the forward and inverse wavelet transforms and R_ℓ acts on the wavelet coefficients. PI-WNO therefore provides a representative transform-basis architecture complementary to PI-DeepONet. Since its prediction is represented on a discrete grid rather than as an explicit coordinate-dependent neural function, the spatial derivatives entering the physics loss are evaluated using the stochastic projection-based gradient estimator adopted by PI-WNO rather than automatic differentiation with respect to the output coordinates. PI-WNO is used as a reference for the parametric Burgers and two-dimensional Poisson benchmarks.

Thus, PI-DeepONet and PI-WNO solve the same physics-informed operator-learning problem without paired solution supervision, but realize the operator through two different basis mechanisms: PI-DeepONet constructs the solution directly from a globally learned output basis, whereas PI-WNO uses a prescribed wavelet basis to parameterize transformations within the operator layers.

C Detailed mathematical formulation of KAPI-ONet

Let $\mathcal{T} \in \mathcal{Q}$ denote a task instance, where $\mathcal{Q}$ is the task space. For computer-vision problems, $\mathcal{T}$ may be an image or an image together with auxiliary task information, such as a point annotation. For PDE problems, $\mathcal{T}$ may be specified by a forcing function, physical parameters, initial or boundary conditions, sparse observations, or a combination of these quantities.

KAPI-ONet associates each task $\mathcal{T}$ with a finite-dimensional task representation $\mathbf{r}(\mathcal{T}) \in \mathbb{R}^{d_r}$, where d_r denotes the dimension of the representation supplied to the conditioning network. The purpose of $\mathbf{r}(\mathcal{T})$ is to express the task in a form that is well suited to constructing the corresponding output field. For coordinate-resolved inputs, a useful choice is a finite set of kernel parameters defining a continuous representation in the physical coordinate system. This is particularly natural for KAPI-ONet because the output field is itself represented by localized kernels whose centers and scales are defined in the computational domain. Using kernel-based representations on both sides therefore provides a common geometric description of the structures present in the task and output fields in terms of coordinate-aware quantities such as locations and scales, and simplifies the

mapping between their respective representations. When the available task information is already low-dimensional, its observed values can instead be used directly as $\mathbf{r}(\mathcal{T})$.

The central operation in KAPI-ONet is a learned map from the task representation to the parameters of a task-dependent kernel field. We denote this representation-conditioning map by H_ψ, where ψ collects its trainable parameters, and write

$$H_\psi : \mathbf{r}(\mathcal{T}) \longmapsto \Theta(\mathcal{T}). \tag{C.1}$$

Here $\Theta(\mathcal{T})$ denotes the kernel state associated with the task. The entries of $\Theta(\mathcal{T})$ determine the geometry and expansion parameters required to construct the corresponding continuous output representation.

Let $\mathcal{D} \subset \mathbb{R}^{d_c}$ denote the domain on which the output field is defined, where d_c is the coordinate dimension, and let $\mathbf{z} \in \mathcal{D}$ denote a query coordinate. Given the task-dependent state $\Theta(\mathcal{T})$, the predicted continuous field is written as

$$\widehat{\mathbf{f}}(\mathbf{z}; \mathcal{T}) = \mathcal{K}\left(\mathbf{z}; \Theta(\mathcal{T})\right), \qquad \mathbf{z} \in \mathcal{D}, \tag{C.2}$$

where $\mathcal{K}$ denotes the kernel-based field representation and $\widehat{\mathbf{f}}$ denotes the predicted output field. Importantly, $\Theta(\mathcal{T})$ is generated from the task representation independently of the query coordinate. Once the task-dependent state has been constructed, the same continuous representation can therefore be evaluated at arbitrary locations in $\mathcal{D}$.

The general KAPI-ONet construction can consequently be summarized as

$$\mathcal{T} \longmapsto \mathbf{r}(\mathcal{T}) \xrightarrow{H_\psi} \Theta(\mathcal{T}) \longmapsto \widehat{\mathbf{f}}(\cdot; \mathcal{T}). \tag{C.3}$$

The defining principle is therefore the task-conditioned construction of an explicit continuous kernel representation of the output field. The mechanism used to represent the task, the parameterization used for $\Theta(\mathcal{T})$, and the objective used to train the resulting field are problem-dependent choices rather than part of this abstract definition.

C.1 Canonical PDE operator-learning realization

We now specialize the abstract construction to function-to-function PDE operator learning, which provides the canonical realization of KAPI-ONet used throughout the following development. Let $v \in \mathcal{V}$ denote an input function, where $\mathcal{V}$ is the admissible input-function space, and let $u \in \mathcal{U}$ denote the corresponding solution field, where $\mathcal{U}$ is the solution space. In this setting, a task instance is identified by its input function, so that $\mathcal{T} = v$, and KAPI-ONet seeks to learn the operator that maps each admissible input function to its corresponding continuous solution field.

Let $\mathcal{D} \subset \mathbb{R}^{d_c}$ denote the computational domain and $\mathbf{z} \in \mathcal{D}$ a computational coordinate. We consider a family of PDE problems written abstractly as

$$\mathcal{N}[v, u](\mathbf{z}) = 0, \qquad \mathbf{z} \in \mathcal{D}, \tag{C.4}$$

together with prescribed boundary conditions $\mathcal{B}[v, u] = 0$ and, when applicable, initial conditions $\mathcal{I}[v, u] = 0$. Here $\mathcal{N}$ denotes the differential operator, while $\mathcal{B}$ and $\mathcal{I}$ denote the boundary- and initial-condition operators, respectively. Assuming that each admissible input v determines a unique solution u, the PDE family defines a solution operator

$$\mathcal{G} : \mathcal{V} \longrightarrow \mathcal{U}, \qquad v \longmapsto u. \tag{C.5}$$

KAPI-ONet approximates this operator by constructing, for each input function v, a task-dependent continuous kernel representation of the corresponding solution field.

C.1.1 Gaussian-splat functional representation

The input function v is available numerically through its values at a finite set of sensor locations. Let $\Omega_v \subset \mathbb{R}^{d_v}$ denote the domain of the input function and let $\{\mathbf{x}_i\}_{i=1}^{N_s} \subset \Omega_v$ denote N_s sensor locations. We collect the corresponding observations through the sampling operator P_s as

$$P_s v = \mathbf{v}_s = [v(\mathbf{x}_1), \ldots, v(\mathbf{x}_{N_s})]^\top \in \mathbb{R}^{N_s}. \tag{C.6}$$

Rather than using the sampled values $\mathbf{v}_s$ directly as the task representation, we construct a continuous kernel representation of the input function in its physical coordinate system. This provides the conditioner with an explicit description of where structures occur in the input and over what coordinate scales they vary.

We obtain this representation using a Gaussian-splat functional autoencoder (GSFAE) illustrated in Fig. C.1. Let E_ϕ denote the trainable encoder with parameters ϕ. It maps the sampled input $\mathbf{v}_s$ to the structured Gaussian-splat state

$$\Theta_v^{\mathrm{GS}} = E_\phi(\mathbf{v}_s) = \left\{a_j^v, \boldsymbol{\mu}_j^v, \sigma_j^v\right\}_{j=1}^{M_v}, \tag{C.7}$$

where M_v denotes the number of input kernels, $a_j^v \in \mathbb{R}$ is the amplitude of the jth kernel, $\boldsymbol{\mu}_j^v \in \mathbb{R}^{d_v}$ is its center, and $\sigma_j^v > 0$ is its scale. The state Θ_v^{GS} therefore has dimension $d_{\mathrm{GS}} = M_v(d_v + 2)$. Its entries retain an explicit geometric interpretation in the coordinate system of the input function.

The structured state directly parameterizes a continuous approximation of v through a fixed analytic Gaussian decoder $\mathcal{D}_{\mathrm{GS}}$. Specifically,

$$\widehat{v}(\mathbf{x}) = \mathcal{D}_{\mathrm{GS}}\left(\Theta_v^{\mathrm{GS}}; \mathbf{x}\right) = \sum_{j=1}^{M_v} a_j^v \exp\left[-\frac{\|\mathbf{x} - \boldsymbol{\mu}_j^v\|_2^2}{2(\sigma_j^v)^2}\right], \qquad \mathbf{x} \in \Omega_v. \tag{C.8}$$

The decoder contains no trainable parameters: the amplitudes, centers, and scales produced by E_ϕ completely determine the reconstructed function. Consequently, Θ_v^{GS} is not an abstract latent code but the parameter set of an explicit continuous kernel representation of the input.

The encoder is pretrained over the input-function family by minimizing the relative reconstruction error

$$\mathcal{L}_{\mathrm{rec}}(\phi) = \mathbb{E}_{v\sim\mu_{\mathcal{V}}}\left[\frac{\left\|v - \mathcal{D}_{\mathrm{GS}}\big(E_\phi(P_s v)\big)\right\|_{L^2(\Omega_v)}^2}{\|v\|_{L^2(\Omega_v)}^2 + \varepsilon}\right], \tag{C.9}$$

where $\mu_{\mathcal{V}}$ denotes the distribution of input functions and $\varepsilon > 0$ is a small constant used for numerical stability. Since $\mathcal{D}_{\mathrm{GS}}$ is fixed, minimizing $\mathcal{L}_{\mathrm{rec}}$ updates only the encoder parameters ϕ.

After pretraining, we denote the resulting encoder parameters by $\phi^\star$ and freeze $E_{\phi^\star}$ during operator learning. For each input function v, the frozen encoder then produces Θ_v^{GS}, which serves as the task representation $\mathbf{r}(\mathcal{T})$ introduced in the general formulation. Since $\mathcal{T} = v$ in the present PDE setting, the next stage of KAPI-ONet learns to map this structured representation of the input function to a structured kernel representation of its corresponding solution.

C.1.2 Task conditioning

After GSFAE pretraining, the frozen encoder $E_{\phi^\star}$ maps each input function v to the structured state Θ_v^{GS}. In the notation of the general framework, this state provides the task representation $\mathbf{r}(\mathcal{T})$ for the present PDE setting. KAPI-ONet then learns a representation-conditioning map from the input-function state to a task-dependent solution state,

$$H_\psi : \Theta_v^{\mathrm{GS}} \longmapsto \Theta_u^{\mathrm{KAPI}}, \tag{C.10}$$

where H_ψ denotes the conditioning network with trainable parameters ψ, and Θ_u^{KAPI} denotes the finite-dimensional state that parameterizes the continuous solution representation associated with v.

The role of H_ψ is to transfer the geometric and functional information contained in the input representation to the kernel representation of the solution. Since different input functions may produce solution features at different locations and coordinate scales, the solution representation is generated separately for each task rather than using a single fixed kernel configuration. Importantly, H_ψ depends on the task representation but not on the query coordinate $\mathbf{z}$. The complete solution state is therefore generated once for a given input function and can subsequently be used to evaluate the continuous solution at arbitrary points in the computational domain.

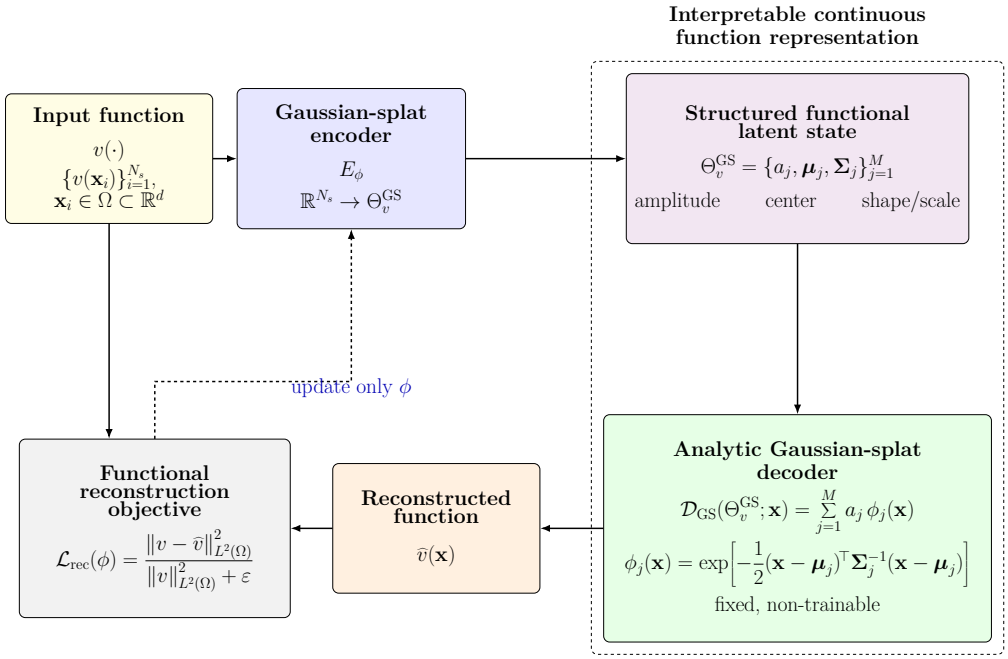

Figure C.1: **Gaussian-splat functional autoencoder.** Sensor observations of the input function $v(\cdot)$ are mapped by the trainable encoder E_ϕ to the structured Gaussian-splat state $\Theta_v^{\mathrm{GS}} = \{a_j^v, \boldsymbol{\mu}_j^v, \sigma_j^v\}_{j=1}^{M_v}$. The state directly parameterizes a continuous approximation of the input function through a fixed, non-trainable analytic Gaussian decoder. The original and reconstructed functions define the functional reconstruction objective $\mathcal{L}_{\mathrm{rec}}$, whose gradients update only the encoder parameters ϕ. After pretraining, $E_{\phi^\star}$ is frozen and supplies the structured functional representation used by the subsequent operator model.

C.1.3 Hierarchical solution representation

For the canonical PDE realization, the conditioner does not generate the geometry of every physical RBF independently. Instead, it produces a hierarchical solution state

$$\Theta_u^{\mathrm{KAPI}}(v) = \left(\Theta_u^{\mathrm{par}}(v), \boldsymbol{\beta}_u(v)\right), \tag{C.11}$$

where $\Theta_u^{\mathrm{par}}(v)$ contains a set of task-dependent parent geometries and $\boldsymbol{\beta}_u(v)$ contains the expansion coefficients of the physical RBFs together with a scalar bias. For the two-dimensional computational coordinates considered here, each parent is described by a center $\boldsymbol{\mu}_k(v) \in \mathbb{R}^2$, two principal scales $s_{k,1}(v), s_{k,2}(v) > 0$, and an orientation $\theta_k(v)$. With K parents, the parent state is therefore

$$\Theta_u^{\mathrm{par}}(v) = \{\boldsymbol{\mu}_k(v), s_{k,1}(v), s_{k,2}(v), \theta_k(v)\}_{k=1}^{K}. \tag{C.12}$$

The task-dependent parents are constructed relative to a shared reference configuration. Let $\mathbf{q}^0 = \{\mathbf{q}_k^0\}_{k=1}^K$ denote this configuration, with $\mathbf{q}_k^0 = (\boldsymbol{\mu}_k^0, s_{k,1}^0, s_{k,2}^0, \theta_k^0)$. The conditioner predicts task-dependent modifications $\Delta\mathbf{q}_k(v)$, from which the parent geometry is obtained through a bounded transformation

$$\mathbf{q}_k(v) = \mathcal{B}\left(\mathbf{q}_k^0, \Delta\mathbf{q}_k(v)\right), \tag{C.13}$$

where $\mathcal{B}$ denotes the transformation used to keep the parent centers, scales, and orientations within their prescribed admissible ranges. The reference configuration is shared across tasks, whereas the modifications produced by H_ψ adapt the parent geometry to the particular input function v.

Each parent generates M physical child RBFs, giving a total of $N = KM$ physical basis functions. To define this construction, let $\boldsymbol{\xi}_{km} \in \mathbb{R}^2$ denote a fixed dimensionless reference coordinate and

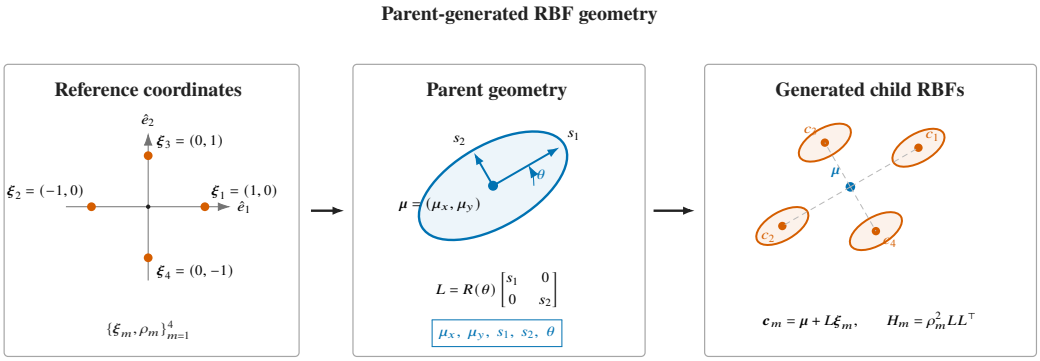

Figure C.2: **Parent-generated RBF geometry.** Illustration of the fixed geometry decoder for a single parent with $M = 4$ children. The parent is described by its center $\boldsymbol{\mu}$, principal scales (s_1, s_2), and orientation θ, defining $L = R(\theta)\,\mathrm{diag}(s_1, s_2)$. Fixed dimensionless reference coordinates $\{\boldsymbol{\xi}_m\}_{m=1}^4$ and relative scales $\{\rho_m\}_{m=1}^4$ are transformed into physical child centers and anisotropic shape matrices through $\mathbf{c}_m = \boldsymbol{\mu} + L\boldsymbol{\xi}_m$ and $H_m = \rho_m^2 LL^\top$. The parent itself is not a physical basis function; only the generated children enter the continuous RBF expansion, and their expansion coefficients remain independent.

let $\rho_{km} > 0$ denote a fixed relative scale for child (k, m). For parent k, define the task-dependent geometric transformation

$$L_k(v) = R(\theta_k(v))\,\mathrm{diag}\left(s_{k,1}(v), s_{k,2}(v)\right), \tag{C.14}$$

where

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{C.15}$$

is the two-dimensional rotation matrix. The center and anisotropic shape matrix of child (k, m) are then

$$\mathbf{c}_{km}(v) = \boldsymbol{\mu}_k(v) + L_k(v)\boldsymbol{\xi}_{km}, \qquad H_{km}(v) = \rho_{km}^2 L_k(v) L_k(v)^\top. \tag{C.16}$$

The collection $\theta_D = \{\boldsymbol{\xi}_{km}, \rho_{km}\}_{k,m}$ constitutes a fixed geometry-decoder template. The corresponding decoder acts only on the parent state,

$$G_{\theta_D} : \Theta_u^{\mathrm{par}}(v) \longmapsto \{\mathbf{c}_{km}(v), H_{km}(v)\}_{k,m}. \tag{C.17}$$

Figure C.2 illustrates this construction for a single parent with four children. The parent determines a task-dependent translation, anisotropic scaling, and rotation, whereas the fixed reference coordinates and relative scales determine the arrangement of its children within that local coordinate system.

The parent geometries themselves are not basis functions. They act only as geometry generators for the physical children. Moreover, sharing the geometry of a parent does not imply sharing the expansion coefficients of its children. Each child retains an independent coefficient $a_{km}(v)$, and the coefficient state is

$$\boldsymbol{\beta}_u(v) = (\{a_{km}(v)\}_{k=1,\ldots,K;\, m=1,\ldots,M}, b(v)) \in \mathbb{R}^{N+1}, \tag{C.18}$$

where $b(v)$ denotes the task-dependent bias. The fixed geometry decoder therefore modifies only the geometric component of Θ_u^{KAPI}, while the coefficients pass through unchanged,

$$G_{\theta_D} : (\Theta_u^{\mathrm{par}}(v), \boldsymbol{\beta}_u(v)) \longmapsto (\{\mathbf{c}_{km}(v), H_{km}(v)\}_{k,m}, \boldsymbol{\beta}_u(v)). \tag{C.19}$$

The parent–child construction is introduced to share geometric degrees of freedom without reducing the number of physical basis functions. In two computational dimensions, an independently parameterized anisotropic Gaussian requires five geometric quantities: two center coordinates, two principal scales, and one orientation. A dictionary of N independently generated RBFs would therefore require $5N$ task-dependent geometric quantities. The hierarchical construction generates only

the five geometric quantities of each of the K parents, requiring $5K$ quantities. Since $N = KM$, the fraction of independently generated geometry is

$$\frac{5K}{5N} = \frac{1}{M}, \tag{C.20}$$

corresponding to a reduction of $1 - 1/M$ in the number of independently generated geometric variables. All N physical RBFs nevertheless remain in the representation and retain independent expansion coefficients.

For a scalar solution field, the conditioner therefore generates $5K+N+1$ task-dependent quantities: $5K$ parent-geometry modifications, N physical-RBF coefficients, and one bias. By comparison, independently parameterizing the geometry and coefficient of every anisotropic RBF would require $6N + 1$ quantities. The parent–child factorization thus provides a direct trade-off between geometric sharing and task-dependent flexibility: smaller K imposes stronger sharing among physical kernels and reduces the dimension of the generated solution state, whereas larger K permits finer independent adaptation of the RBF geometry.

C.1.4 Continuous solution representation and exact constraints

Once the physical child geometries $\{\mathbf{c}_{km}(v), H_{km}(v)\}_{k,m}$ have been generated, each child defines a Gaussian basis function over the computational domain,

$$\phi_{km}(\mathbf{z}; v) = \exp\left[-\frac{1}{2}\left(\mathbf{z} - \mathbf{c}_{km}(v)\right)^{\top} H_{km}(v)^{-1}\left(\mathbf{z} - \mathbf{c}_{km}(v)\right)\right], \tag{C.21}$$

where $\mathbf{z} \in \mathcal{D}$ is the query coordinate. Together with the task-dependent coefficients $\{a_{km}(v)\}_{k,m}$ and bias $b(v)$ contained in $\boldsymbol{\beta}_u(v)$, these basis functions define the unconstrained continuous solution representation

$$\widetilde{u}(\mathbf{z}; v) = b(v) + \sum_{k=1}^{K}\sum_{m=1}^{M} a_{km}(v)\phi_{km}(\mathbf{z}; v). \tag{C.22}$$

Thus, the conditioner determines the complete parametric representation of the solution from the input function, while the query coordinate enters only when the resulting field is evaluated. The same task-dependent state can therefore be reused to evaluate $\widetilde{u}(\mathbf{z}; v)$ at arbitrary locations in $\mathcal{D}$ without regenerating the RBF geometry or coefficients.

When prescribed boundary or initial conditions admit a suitable analytic construction, they can be incorporated directly into the solution representation rather than enforced only through residual penalties. We write the constrained trial solution as

$$\widehat{u}(\mathbf{z}; v) = A_v(\mathbf{z}) + B_v(\mathbf{z})\widetilde{u}(\mathbf{z}; v), \tag{C.23}$$

where $A_v(\mathbf{z})$ is chosen to satisfy the non-homogeneous part of the prescribed constraints and $B_v(\mathbf{z})$ is constructed to vanish on the corresponding boundary and/or initial-condition manifold. The subscript v allows either function to depend on the task when the prescribed constraints themselves vary with the input.

Substituting Eq. equation C.22 gives

$$\widehat{u}(\mathbf{z}; v) = A_v(\mathbf{z}) + B_v(\mathbf{z})\left[b(v) + \sum_{k=1}^{K}\sum_{m=1}^{M} a_{km}(v)\phi_{km}(\mathbf{z}; v)\right]. \tag{C.24}$$

This construction separates the known analytical structure of the problem from the task-dependent component learned by KAPI-ONet. Constraints embedded through A_v and B_v are satisfied identically for every task and query point, allowing the physics-informed optimization to focus on the remaining governing equations and on any constraints that cannot be imposed analytically.

C.1.5 Physics-informed operator training

The task-dependent solution representation in Eq. equation C.24 is trained by requiring it to satisfy the governing PDE over a distribution of input functions. For an input $v \in \mathcal{V}$, the interior physics

residual is obtained by substituting the predicted field $\widehat{u}(\cdot; v)$ into the differential operator introduced in Eq. equation C.4,

$$r_{\mathrm{PDE}}(\mathbf{z}; v) = \mathcal{N}\left[v, \widehat{u}(\cdot; v)\right](\mathbf{z}). \tag{C.25}$$

Let $\{\mathbf{z}_i^f\}_{i=1}^{N_f} \subset \mathcal{D}$ denote a set of N_f interior collocation points. The corresponding physics residual loss is

$$\mathcal{L}_{\mathrm{PDE}}(v) = \frac{1}{N_f} \sum_{i=1}^{N_f} \left| r_{\mathrm{PDE}}\left(\mathbf{z}_i^f; v\right) \right|^2. \tag{C.26}$$

For systems of equations, the same construction is applied componentwise and the squared residuals are accumulated over the governing equations.

Boundary and initial conditions that are embedded analytically through A_v and B_v in Eq. equation C.23 are satisfied identically and require no additional penalty. When a prescribed constraint cannot or is not imposed through the trial representation, we denote its corresponding residual losses by $\mathcal{L}_{\mathrm{BC}}(v)$ and $\mathcal{L}_{\mathrm{IC}}(v)$ for boundary and initial conditions, respectively. Additional weak penalties used to control the kernel representation are collected in $\mathcal{L}_{\mathrm{reg}}(v)$.

The physics-informed objective for an input task is therefore

$$\mathcal{L}(v) = w_{\mathrm{PDE}}\mathcal{L}_{\mathrm{PDE}}(v) + w_{\mathrm{BC}}\mathcal{L}_{\mathrm{BC}}(v) + w_{\mathrm{IC}}\mathcal{L}_{\mathrm{IC}}(v) + w_{\mathrm{reg}}\mathcal{L}_{\mathrm{reg}}(v), \tag{C.27}$$

where w_{PDE}, w_{BC}, w_{IC}, and w_{reg} are non-negative weights. Terms corresponding to constraints that are imposed exactly, or that are not present in a given problem, are omitted.

Let ϑ denote the parameters optimized during operator training. In the hierarchical PDE realization, these include the conditioner parameters ψ, the shared reference parent configuration, and any shared trainable scales used in the kernel parameterization. The pretrained GSFAE parameters $\phi^\star$ and the fixed geometry-decoder template θ_D remain frozen. The operator-learning problem is then

$$\vartheta^\star = \arg\min_{\vartheta} \; \mathbb{E}_{v\sim\mu_{\mathcal{V}}}\left[\mathcal{L}(v)\right], \tag{C.28}$$

where $\mu_{\mathcal{V}}$ denotes the distribution of input functions used for operator training. Please note that the physics-informed formulation is not restricted to minimizing residuals of the governing differential equations. When the problem admits a variational description, the task-conditioned field may instead be trained by minimizing the corresponding energy functional directly (Goswami et al., 2022; Dwivedi et al., 2024).

This training procedure requires the input functions and the governing equations, but not paired numerical solution fields. Each sampled input generates its own kernel geometry and expansion coefficients through H_ψ, and the same task-dependent continuous field is evaluated at the collocation points required to form the physics residuals. KAPI-ONet therefore learns the map from input-function representations to solution representations directly through the governing physics rather than through supervision by precomputed solution values.

Combining the preceding components, the canonical KAPI-ONet realization for function-to-function PDE operator learning follows the sequence

$$v \longrightarrow \mathbf{v}_s \xrightarrow{E_{\phi^\star}} \Theta_v^{\mathrm{GS}} \xrightarrow{H_\psi} \Theta_u^{\mathrm{KAPI}} = (\Theta_u^{\mathrm{par}}, \boldsymbol{\beta}_u) \xrightarrow{G_{\theta_D}} (\{\mathbf{c}_{km}, H_{km}\}_{k,m}, \boldsymbol{\beta}_u) \longrightarrow \widehat{u}(\cdot; v). \tag{C.29}$$

The sampled input function is first converted into a continuous structured kernel representation, from which the conditioner generates a task-dependent solution state. The fixed geometry decoder then expands the parent geometry into the physical RBF dictionary, while the independently generated coefficients are retained. The resulting continuous solution field is finally constrained by the governing equations during operator training. Figure C.3 summarizes this complete computational and training flow.

D TEST CASES

We evaluate KAPI-ONet on a sequence of problems of increasing task and physical complexity. The first three PDE families assess forcing-to-solution and initial-condition-to-solution maps for linear and nonlinear steady and time-dependent equations. We then consider the steady incompressible Navier–Stokes equations as a finite-dimensional parameter-to-field problem, which serves as

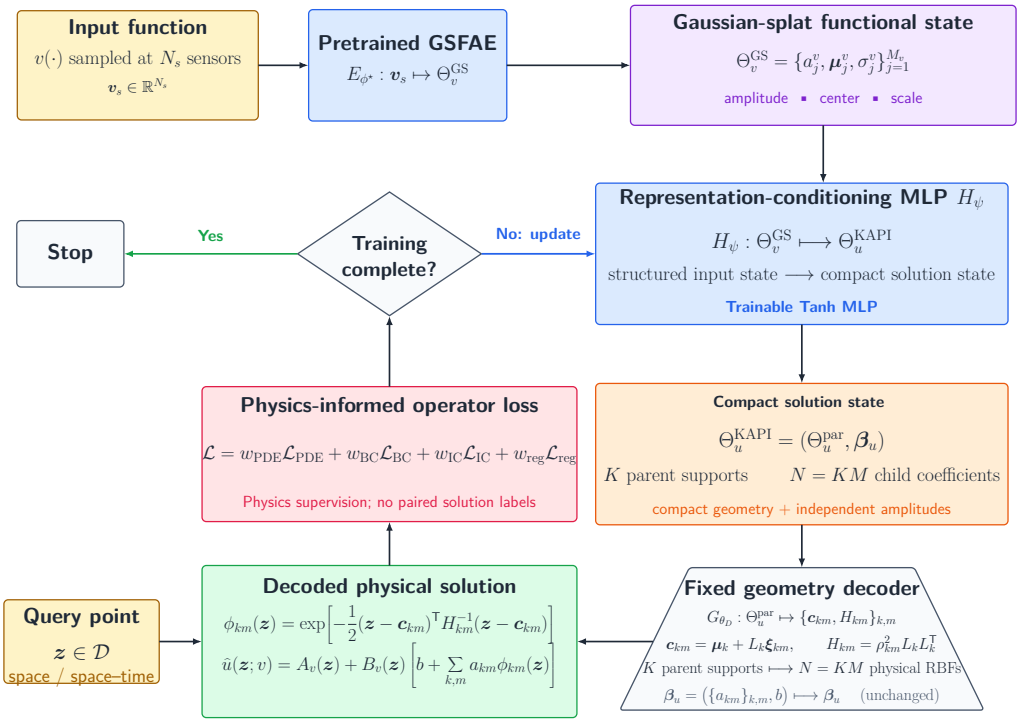

Figure C.3: **Canonical KAPI-ONet architecture for functional-input PDE operator learning.** An input function $v(\cdot)$, sampled at N_s sensors, is mapped by the pretrained GSFAE to the structured functional state Θ_v^{GS}. The representation-conditioning network H_ψ maps this state to the task-dependent solution state $\Theta_u^{\mathrm{KAPI}} = (\Theta_u^{\mathrm{par}}, \boldsymbol{\beta}_u)$. The fixed geometry decoder expands the K parent supports into $N = KM$ physical Gaussian RBFs while leaving the independently generated expansion coefficients and bias unchanged. The resulting continuous solution is evaluated at arbitrary query coordinates and constrained during training by the governing PDE, boundary and initial conditions when required, and representation regularization.

a bridge between operator and parametric learning. Finally, we consider two application-driven settings: sparse-observation inverse hemodynamics using synthetic flow data, where the principal quantity of interest is wall shear stress, and weakly supervised cardiac annotation using real medical images, where per-image variational annotation is extended to operator learning.

1. **2D non-homogeneous Poisson equation.**
   We consider two source-to-solution operator-learning settings for the two-dimensional Poisson equation.
   Case A: Localized Gaussian sources. We solve

$$-\Delta u(x,y) = f(x,y), \qquad (x,y) \in (0,1)^2,$$

   subject to homogeneous Dirichlet boundary conditions,

$$u|_{\partial\Omega} = 0.$$

   The source field is composed of four localized Gaussian functions (Dwivedi et al., 2026),

$$f(x,y) = \sum_{j=1}^{4} a_j \exp\left[-\frac{(x-\mu_{x,j})^2 + (y-\mu_{y,j})^2}{2\sigma_j^2}\right],$$

   where a_j, $(\mu_{x,j}, \mu_{y,j})$, and σ_j denote the amplitude, center, and width of the jth source, respectively.

Each forcing realization is observed on a 64×64 spatial grid. Performance is evaluated against independently computed finite-difference reference solutions.

Case B: Analytical Poisson benchmark. We additionally consider

$$\Delta u(x,y) = f(x,y), \qquad (x,y) \in (-1,1)^2,$$

with

$$u|_{\partial\Omega} = 0.$$

Following the corresponding PI-WNO benchmark (N. et al., 2024), the solution family is

$$u(x,y) = \alpha \sin(\pi x)\big(1+\cos(\pi y)\big) + \beta \sin(2\pi x)\big(1-\cos(2\pi y)\big),$$

where

$$\alpha, \beta \sim \mathcal{U}(-2,2).$$

The corresponding forcing is obtained analytically as

$$f(x,y) = -\alpha\pi^2 \sin(\pi x)\big(1+2\cos(\pi y)\big) + 4\beta\pi^2 \sin(2\pi x)\big(2\cos(2\pi y)-1\big).$$

Each forcing realization is observed on a 65×65 spatial grid. The analytical solution is used only for post-training evaluation.

For both cases, the target operator is

$$\mathcal{G} : f(x,y) \mapsto u(x,y).$$

2. **1D diffusion–reaction equation.**

We consider the nonlinear diffusion–reaction benchmark used in DeepONet and physics-informed DeepONet studies (Lu et al., 2021; Wang et al., 2021b),

$$\frac{\partial s}{\partial t} = 0.01\frac{\partial^2 s}{\partial x^2} + 0.01 s^2 + u(x), \qquad (x,t) \in (0,1) \times (0,1].$$

We impose homogeneous Dirichlet boundary conditions,

$$s(0,t) = s(1,t) = 0, \qquad t \in [0,1],$$

and the zero initial condition,

$$s(x,0) = 0, \qquad x \in [0,1].$$

The operator input is the forcing function $u(x)$, sampled from a zero-mean Gaussian random field,

$$u \sim \mathcal{GP}(0,K),$$

with squared-exponential covariance kernel

$$K(x,x') = \exp\left(-\frac{(x-x')^2}{2\ell^2}\right), \qquad \ell = 0.2.$$

Each forcing realization is observed at 100 uniformly spaced spatial locations.

The target operator is

$$\mathcal{G} : u(x) \mapsto s(x,t).$$

KAPI-ONet is trained without paired solution fields. Performance is evaluated against independently computed finite-difference reference solutions using the relative L^2 error.

3. **1D viscous Burgers equation.**

We consider

$$u_t + u\,u_x = \nu\,u_{xx}, \qquad (x,t) \in (0,1) \times (0,1],$$

under two input distributions (N. et al., 2024; Wang et al., 2021b).

Case A: Parametric IC. We set $\nu = 0.1$ and impose homogeneous Dirichlet boundary conditions,

$$u(0,t) = u(1,t) = 0.$$

The initial condition is

$$u(x,0) = 4x(1-x)\left[\cos(\zeta\pi x) + \sin(\eta\pi x)\right], \qquad \zeta, \eta \overset{\text{iid}}{\sim} \mathcal{U}(0.5, 1.5).$$

Each initial condition is observed at 100 uniformly spaced spatial sensors.

Case B: GRF IC. We set $\nu = 0.01$ and impose periodic boundary conditions,

$$u(0,t) = u(1,t), \qquad u_x(0,t) = u_x(1,t).$$

The initial condition is sampled from the Gaussian random field

$$u(x,0) = u_0(x), \qquad u_0 \sim \mathcal{N}\big(0,\, 25^2(-\Delta + 5^2 I)^{-4}\big)\,.$$

Each realization is represented at 101 uniformly spaced spatial locations.

For both cases, the target operator is

$$\mathcal{G} : u_0(x) \mapsto u(x,t).$$

4. **2D steady incompressible Navier–Stokes equations.**

We consider the two-dimensional lid-driven cavity on

$$\Omega = (0,1)^2.$$

The steady velocity field $\mathbf{u} = (u, v)$ and pressure p satisfy

$$\nabla \cdot \mathbf{u} = 0,$$

and

$$\rho(\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \mu \Delta \mathbf{u}, \qquad (x,y) \in \Omega.$$

We set $\rho = 1$, $U_{\text{lid}} = 1$, and $L = 1$, and parameterize the viscosity through $Re = \rho U_{\text{lid}} L/\mu = 1/\mu$ over $Re \in [10, 100]$, so that $\mu = 1/Re$.

No-slip conditions are imposed on the left, right, and bottom walls,

$$u(0,y) = v(0,y) = 0, \qquad u(1,y) = v(1,y) = 0, \qquad 0 \le y < 1,$$

and

$$u(x,0) = v(x,0) = 0, \qquad 0 \le x \le 1.$$

The upper wall moves with unit horizontal velocity,

$$u(x,1) = 1, \qquad v(x,1) = 0, \qquad 0 < x < 1.$$

The top corner values are inherited from the stationary side walls,

$$u(0,1) = u(1,1) = 0, \qquad v(0,1) = v(1,1) = 0.$$

Since the pressure is defined up to an additive constant, we impose

$$\int_\Omega p(x,y)\, dx\, dy = 0.$$

The operator input is the Reynolds number, and the target parameter-to-field map is

$$\mathcal{G}_{\text{par}} : Re \mapsto \big(u(\cdot,\cdot;Re), v(\cdot,\cdot;Re), p(\cdot,\cdot;Re)\big), \qquad Re \in [10, 100].$$

5. **Sparse-observation inverse hemodynamics and wall shear stress.**

We consider a clinically motivated inverse-flow problem inspired by Arzani et al. (2021), in which near-wall hemodynamics must be reconstructed from sparse velocity measurements when the flow boundary conditions are not fully known. Unlike Arzani et al. (2021), which solves an inverse problem for a single flow realization using a PINN, we extend this setting to operator learning over a family of stenotic flows.

The computational domain is the two-dimensional symmetric stenotic channel

$$\Omega = \{(x,y) : 0 < x < 1,\ -h(x) < y < h(x)\}\,,$$

where

$$h(x) = \begin{cases} 0.05, & 0 \le x < 0.2, \\ 0.05 - 0.02\, S\left(\dfrac{x - 0.2}{0.3}\right), & 0.2 \le x \le 0.5, \\ 0.03 + 0.02\, S\left(\dfrac{x - 0.5}{0.3}\right), & 0.5 < x \le 0.8, \\ 0.05, & 0.8 < x \le 1, \end{cases} \qquad S(\xi) = 3\xi^2 - 2\xi^3.$$

The steady velocity field $\mathbf{u} = (u, v)$ and pressure p satisfy

$$\nabla \cdot \mathbf{u} = 0, \qquad \rho(\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \mu \Delta \mathbf{u},$$

with $\rho = 1$ and $\mu = 5 \times 10^{-5}$. For generation of the hidden FEM references, no-slip conditions are imposed on both walls, the inlet profile is

$$\mathbf{u}(0, y) = \left(U_c \left[1 - \left(\frac{y}{R} \right)^2 \right], 0 \right), \qquad R = 0.05,$$

and the outlet reference pressure is $p(1, y) = 0$. The task-dependent inlet amplitude is sampled as $U_c \sim \mathcal{U}(0.01, 0.10)$, giving $Re = \rho U_c R/\mu \in [10, 100]$.

The wall geometry and no-slip behavior are known to the learned model, whereas the task-specific inlet profile, U_c, Reynolds number, outlet pressure condition, pressure field, flow rate, and complete FEM velocity field are not supplied as operator inputs. For each realization, the only measured quantities provided to the conditioner are the two velocity components at five fixed interior locations,

$$(0.50, 0.000), \quad (0.58, 0.016), \quad (0.66, -0.016), \quad (0.74, 0.012), \quad (0.82, -0.012).$$

The corresponding sensor vector is

$$\mathbf{s} = [u_1, v_1, \ldots, u_5, v_5] \in \mathbb{R}^{10},$$

and the inverse operator is

$$\mathcal{G}_{\text{inv}} : \mathbf{s} \mapsto \big(u(x, y), v(x, y), p(x, y) \big).$$

The reconstructed flow is subsequently used to compute wall shear stress. For a Newtonian fluid with Cauchy stress tensor

$$\boldsymbol{\sigma} = -p\mathbf{I} + \mu \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T \right),$$

the wall shear-stress vector is

$$\boldsymbol{\tau}_w = \left(\mathbf{I} - \mathbf{n}\mathbf{n}^T \right) \boldsymbol{\sigma} \mathbf{n}, \qquad \text{WSS} = \|\boldsymbol{\tau}_w\|_2.$$

The primary quantity of interest is WSS in the stenotic and downstream region; full-field velocity and pressure errors are reported as secondary diagnostics.

6. **Weakly supervised variational segmentation.**

We consider a real-world medical image annotation problem motivated by physics-informed contour selection (Dwivedi et al., 2024). Conventional supervised segmentation networks require dense expert masks for training, whose generation can be costly and time-consuming. Variational segmentation methods such as Chan–Vese, in contrast, optimize image-based energy functionals on a per-image basis without requiring dense training labels. Here, we extend this idea from per-image optimization to *operator learning*: a single KAPI-ONet learns to map an image and a weak localization cue to a continuous implicit field without using dense expert masks during training. The resulting LV boundary is obtained from the zero level set of this field.

Figure D.1 summarizes the resulting operator-learning formulation. The essential distinction from conventional supervised segmentation is that the target mask is not supplied as a training label; instead, the operator is learned from an image-based variational objective and a minimal localization cue.

We use the ACDC cardiac MRI dataset (Bernard et al., 2018; Dwivedi et al., 2024), comprising short-axis cine-MRI scans of 100 patients from five pathological groups, with expert manual annotations of the left ventricle (LV), right ventricle, and myocardium at end diastole (ED) and end systole. We consider the ED phase and focus on annotation of the LV cavity.

For each image slice, the available weak annotation is a single point $c_0 \in \mathbb{R}^2$ located inside the LV cavity. The operator input is (I, c_0) and the output is a continuous implicit field $\phi(x, y; I, c_0)$ represented by a task-dependent RBF model. The corresponding LV region is defined by

$$\Omega_{\text{LV}} = \{(x, y) : \phi(x, y; I, c_0) < 0\},$$

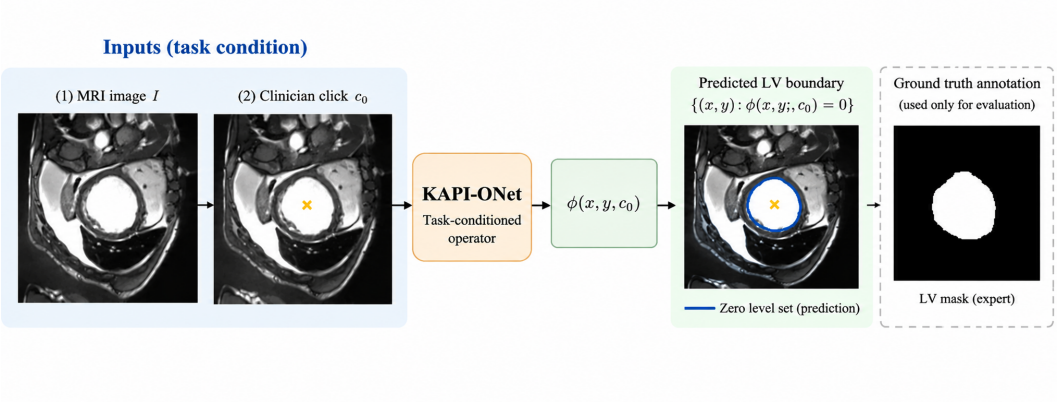

Figure D.1: Weakly supervised cardiac annotation as an operator-learning problem. The task input consists of a cardiac MR image I and a single clinician-provided point c_0 located inside the left-ventricular cavity. KAPI-ONet maps this task condition to a continuous implicit RBF field $\phi(\cdot, \cdot; I, c_0)$, whose zero level set defines the predicted LV boundary. The expert LV mask is not used during operator training and is shown only as a retrospective reference for validation and held-out evaluation.

with boundary

$$\partial\Omega_{\mathrm{LV}} = \{(x, y) : \phi(x, y; I, c_0) = 0\}.$$

The target mapping is therefore

$$\mathcal{G}_{\mathrm{seg}} : (I, c_0) \mapsto \phi(\cdot, \cdot; I, c_0),$$

or equivalently to the corresponding closed LV contour.

The operator is trained using a weakly supervised variational objective containing the Chan–Vese region-based loss (Chan & Vese, 2001) together with weak localization and geometric regularization terms; dense expert segmentation masks are not used for training. Accordingly, this experiment is not intended as a direct accuracy comparison with fully supervised segmentation networks such as U-Net (Azad et al., 2024). Instead, it evaluates whether per-image variational annotation can be extended to operator learning, enabling rapid generation of useful candidate annotations from minimal clinician input and thereby reducing the burden of manual labeling.

E Detailed results

The organization of this section reflects the common structure of the experiments while avoiding repetition. For the first three PDE families (Poisson, diffusion–reaction, and Burgers), the solution is a scalar field represented throughout by the same task-conditioned Gaussian RBF ansatz,

$$\widehat{u}(\mathbf{x}; \mathcal{T}) = b(\mathcal{T}) + \sum_{j=1}^{N} a_j(\mathcal{T}) \varphi_j(\mathbf{x}; \Theta_j(\mathcal{T})),$$

where the task $\mathcal{T}$ determines the RBF geometry and coefficients. Their physics-informed objectives likewise have the common form

$$\mathcal{L} = \lambda_r \mathcal{L}_{\mathrm{PDE}} + \lambda_b \mathcal{L}_{\mathrm{BC}} + \lambda_i \mathcal{L}_{\mathrm{IC}} + \mathcal{L}_{\mathrm{reg}},$$

with terms omitted when not required by the corresponding PDE. We therefore report only the input representation, task conditioning, numerical results, and literature comparison or interpretation for these cases.

The subsequent problems require problem-specific output representations or training objectives and are described explicitly in their corresponding subsections. The lid-driven cavity predicts the coupled primary variables (u, v, p), whereas the inverse-hemodynamics problem uses a streamfunction–pressure representation to impose incompressibility structurally. Their physics-informed objectives

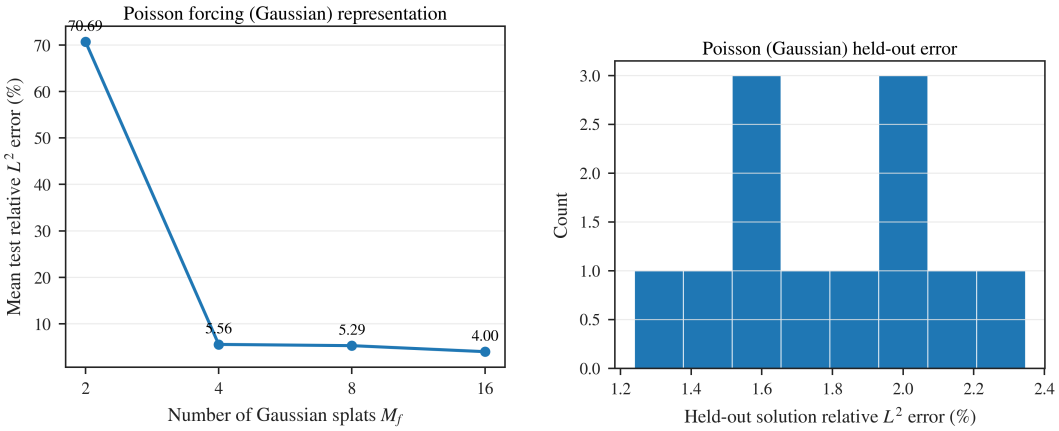

Figure E.1: Localized-Gaussian Poisson results. Left: GSFAE capacity study for the forcing representation, showing a pronounced knee at $M_f = 4$, consistent with the four-component forcing family. Right: distribution of the relative L^2 solution error over the 12 held-out problems used for numerical evaluation.

consequently contain additional boundary, gauge, measurement, or representation-regularization terms. The final cardiac-annotation problem is different again: the task is specified by an image and a clinician-provided localization point, the output is a continuous implicit RBF field whose zero level set defines the annotation, and training minimizes a weakly supervised variational objective rather than a PDE residual. These formulations are therefore given in detail where they are introduced. The architecture details for all the cases are given in F.

E.1 2D NON-HOMOGENEOUS POISSON EQUATION

E.1.1 CASE I: LOCALIZED GAUSSIAN FORCING

- **GSFAE.** We first consider a pedagogical Poisson problem in which each forcing field is generated as a superposition of four localized Gaussian components. The GSFAE is provided only with the resulting two-dimensional forcing fields, without access to this underlying component structure. We consider $M_f \in \{2, 4, 8, 16\}$ Gaussian splats for the forcing representation. As shown in the left panel of Figure E.1, the mean held-out reconstruction error decreases sharply from 70.69% for $M_f = 2$ to 5.56% for $M_f = 4$, followed by only modest improvements to 5.29% and 4.00% for $M_f = 8$ and $M_f = 16$, respectively. The pronounced knee at $M_f = 4$ is consistent with the intrinsic four-component structure of the forcing family. We therefore retain $M_f = 4$ for downstream operator training, giving a compact 16-dimensional Gaussian state.
- **Task conditioning.** The KAPI-ONet operator is trained using 2000 forcing functions. The frozen 16-dimensional forcing state is mapped to $K = 25$ solution parents, with eight children generated per parent, giving $N = 200$ Gaussian RBFs. The conditioner contains 96 259 trainable parameters and is trained for 10 000 updates. Operator training requires 993 s (16.5 min) on CPU.

  Figure E.2 illustrates the resulting mapping from the compact Gaussian representation of the forcing to the task-conditioned Gaussian representation of the solution. Shown are the sampled forcing fields, their Gaussian-splat representations, solution parents and children, predicted solutions, numerical references, and absolute errors for held-out cases at the 10th, 35th, 65th, and 90th error percentiles. The corresponding relative L^2 errors are 1.45%, 1.62%, 1.97%, and 2.15%.
- **Results.** Over the 12 held-out forcing functions used for numerical evaluation, KAPI-ONet achieves a mean relative L^2 error of 1.78%, a median error of 1.75%, and a 90th-percentile error of 2.13%. The maximum observed error is 2.35%, with the complete evaluated distribution shown in the right panel of Figure E.1.

- **Interpretation.** This example is intended as a controlled demonstration of the representation principle underlying KAPI-ONet rather than as a benchmark against a previously published operator-learning model. The GSFAE is given only the sampled two-dimensional forcing fields, yet its capacity sweep exhibits a pronounced knee at $M_f = 4$, consistent with the known four-component construction. The conditioner then maps this compact input geometry to a richer Gaussian representation of the PDE solution.

  The resulting representation is geometrically and physically inspectable: the Gaussian centers, widths, and amplitudes can be visualized directly, and the splats adapt spatially to regions that dominate the solution behavior. In the more complex examples considered subsequently, this adaptation is manifested by clustering near regions of strong variation, alignment with characteristic solution structures, and reduced widths where sharper spatial or space–time gradients must be resolved. The broader role of this explicit geometric parameterization is discussed in Section C.

E.1.2 Case II: Sinusoidal forcing

- **GSFAE.** We consider $M_f \in \{16, 32, 64, 80\}$ Gaussian splats for the forcing representation. As shown in the left panel of Figure E.3, the mean held-out reconstruction error decreases from 25.35% for $M_f = 16$ to 12.23% for $M_f = 32$, 5.61% for $M_f = 64$, and 5.34% for $M_f = 80$. Since the improvement beyond $M_f = 64$ is marginal, we use $M_f = 64$ for downstream operator training, corresponding to a 256-dimensional Gaussian state.
- **Task conditioning.** The KAPI-ONet operator is trained using 500 forcing functions. The frozen 256-dimensional forcing state is mapped to $K = 25$ solution parents, with eight children generated per parent, giving $N = 200$ Gaussian RBFs. The conditioner contains $132\,019$ trainable parameters and is trained for $10\,000$ updates. Operator training requires 1075 s (17.9 min) on CPU, while the complete GSFAE and operator-training pipeline requires approximately 45.8 min.

  Figure E.4 shows the sampled forcing fields, their Gaussian-splat representations, task-conditioned parent and child geometries, predicted solutions, analytical references, and absolute errors for held-out cases at the 10th, 35th, 65th, and 90th error percentiles. The corresponding relative L^2 errors are 0.18%, 0.24%, 0.34%, and 0.56%, respectively.
- **Results.** Over 100 independently held-out forcing functions, KAPI-ONet achieves a mean relative L^2 error of 0.39%, a median error of 0.28%, and a 90th-percentile error of 0.56%. The error distribution is concentrated below 1%, with a small right tail extending to approximately 3%, as shown in the right panel of Figure E.3.
- **Benchmarking.** This experiment closely follows the non-homogeneous Poisson benchmark of PIWNO (N. et al., 2024), using the same 65×65 spatial resolution, 500 training functions, 100 test functions, and sinusoidal parametric solution family. PIWNO reports a relative MSE of $0.0469\% \pm 0.015\%$. Evaluating KAPI-ONet using the metric implemented in the released PI-WNO evaluation code gives $0.366\% \pm 0.401\%$, while its conventional relative L^2 error is $0.387\% \pm 0.407\%$. Thus, although PIWNO attains the lower numerical error, both methods operate in the sub-percent mean-error regime.

  The complete KAPI-ONet pipeline requires approximately 45.8 min on CPU. PIWNO does not report training wall-clock time; however, our same-platform CPU reproduction requires approximately 3.8 min per epoch, corresponding to a projected training time of approximately 25 h for the 400-epoch configuration. The comparison therefore shows a substantial reduction in computational cost for KAPI-ONet while retaining sub-percent prediction accuracy.

E.2 1D diffusion–reaction equation

- **GSFAE.** We consider $M_f \in \{4, 8, 16, 32, 64\}$ Gaussian splats for the forcing representation. As shown in the left panel of Figure E.5, the validation reconstruction error decreases from 8.74% for $M_f = 4$ to 1.35% for $M_f = 32$, after which it increases to 1.72% for $M_f = 64$. We therefore select $M_f = 32$, corresponding to a 96-dimensional Gaussian state. The selected model gives a mean held-out forcing reconstruction error of 1.34%.

- **Task conditioning.** The KAPI-ONet operator is trained using 2000 forcing functions. The frozen 96-dimensional forcing state is mapped to $K = 25$ solution parents, with eight children generated per parent, giving $N = 200$ Gaussian RBFs. The conditioner contains 108 179 trainable parameters and is trained for 10 000 updates. Operator training requires 1358 s (22.6 min) on CPU.

- **Results.** Over 250 independently held-out forcing functions, KAPI-ONet achieves a mean relative L^2 error of 1.00%, a median error of 0.84%, and a 90th-percentile error of 1.68%. The maximum observed error is 5.78%. The corresponding error distribution is shown in the left panel of Figure E.6. The five-model GSFAE capacity sweep requires 908 s (15.1 min); together with operator training, the complete measured training pipeline requires approximately $2266\ \mathrm{s} \approx 37.8\ \mathrm{min}$ on CPU.

- **Benchmarking.** To align the training and test-set sizes with the PI-DeepONet diffusion–reaction benchmark, we repeat the KAPI-ONet experiment using 10 000 operator-training forcing functions and 1000 held-out test functions, matching the training and test-set sizes used by PI-DeepONet. Under this matched setting, the GSFAE selects $M_f = 64$, corresponding to a 192-dimensional forcing state. The corresponding capacity study is shown in the right panel of Figure E.5.

  KAPI-ONet achieves a mean relative L^2 error of 0.76%, a median error of 0.61%, and a 90th-percentile error of 1.32% under the matched configuration. The corresponding distribution is shown in the right panel of Figure E.6. Compared with the original 2000-function experiment, for which the mean error is 1.00%, increasing the number of operator-training functions reduces the mean error by approximately 24%.

  Figure E.7 shows the KAPI-ONet representation for held-out cases at the 10th, 35th, 65th, and 90th error percentiles under the matched configuration. The corresponding relative L^2 errors are 0.33%, 0.50%, 0.77%, and 1.32%, respectively.

  PI-DeepONet reports a mean relative L^2 error of $0.45\% \pm 0.16\%$ over 1000 test functions, compared with $0.76\% \pm 0.56\%$ for the matched KAPI-ONet experiment. PI-DeepONet therefore retains the lower mean solution error on this benchmark. However, the two methods exhibit a substantial difference in training cost. KAPI-ONet requires 10 000 operator-training updates and contains 122 483 trainable conditioner parameters, with operator training taking approximately 1024 s (17.1 min) on CPU. Including the complete GSFAE capacity study and operator training, the full KAPI-ONet training pipeline requires less than 40 min on our CPU platform.

  PI-DeepONet, in contrast, uses 120 000 optimization iterations. The original study reports a training time of 2.27 h on a single NVIDIA V100 GPU. To provide a direct wall-clock comparison on the same computational platform used for KAPI-ONet, we additionally reproduced the PI-DeepONet diffusion–reaction training on our CPU system. The 120 000-iteration PI-DeepONet run requires approximately 40 h, compared with less than 40 min for the complete KAPI-ONet pipeline. This corresponds to a reduction of more than $60\times$ in end-to-end wall-clock training time on the same CPU platform. Thus, while PI-DeepONet attains the lower mean solution error, KAPI-ONet provides a substantially lower training cost, revealing a clear accuracy–computational-cost trade-off on this benchmark.

  For additional context, the conventional DeepONet (Lu et al., 2021) reported in the same study achieves a mean relative L^2 error of $1.92\% \pm 1.12\%$ using 10 000 input functions together with 100 paired numerical solution measurements per function. KAPI-ONet achieves a lower reported mean error in both configurations considered here: using only 2000 input functions and no paired solution fields, it achieves a mean error of 1.00%, while the 10 000-function configuration achieves 0.76%. Thus, relative to the reported conventional DeepONet result, KAPI-ONet reduces the mean error by approximately 48% while using one fifth as many input functions in the smaller configuration, and by approximately 60% at the same number of input functions in the matched configuration, without requiring paired solution data in either case.

E.3 1D VISCOUS BURGERS EQUATION

E.3.1 CASE I: PARAMETRIC INITIAL CONDITIONS

- **GSFAE.** We consider $M_f \in \{4, 8, 16, 32, 64\}$ Gaussian splats for the initial-condition representation. As shown in the left panel of Figure E.8, the validation reconstruction error decreases from 1.08% for $M_f = 4$ to 0.16% for $M_f = 32$, after which it increases slightly to 0.17% for $M_f = 64$. We therefore select $M_f = 32$, corresponding to a 96-dimensional Gaussian state. The selected model gives a mean held-out initial-condition reconstruction error of 0.17%.
- **Task conditioning.** The KAPI-ONet operator is trained using 2000 initial-condition functions. The frozen 96-dimensional initial-condition state is mapped to $K = 25$ solution parents, with eight children generated per parent, giving $N = 200$ Gaussian RBFs. The conditioner contains $108\,179$ trainable parameters and is trained for $10\,000$ updates. Operator training requires 1525 s (25.4 min) on CPU.

  Figure E.9 shows the sampled initial conditions, their Gaussian-splat representations, task-conditioned parent and child geometries, predicted solutions, numerical references, and absolute errors for held-out cases at the 10th, 35th, 65th, and 90th error percentiles. The corresponding relative L^2 errors are 0.11%, 0.18%, 0.31%, and 0.53%.
- **Results.** Over 250 independently held-out initial conditions, KAPI-ONet achieves a mean relative L^2 error of 0.28%, a median error of 0.23%, and a 90th-percentile error of 0.53%. The maximum observed error is 1.04%, and the error distribution is concentrated well below 1%, as shown in the right panel of Figure E.8. The five-model GSFAE capacity sweep requires approximately 665 s (11.1 min); together with operator training, the complete measured pipeline requires approximately $2190\ \mathrm{s} \approx 36.5\ \mathrm{min}$ on CPU.
- **Benchmarking.** For comparison, PIWNO (N. et al., 2024) reports a relative MSE of $0.0518\% \pm 0.041\%$ for its parametric Burgers benchmark. Evaluating the KAPI-ONet predictions using the metric implemented in the released PI-WNO evaluation code gives $0.271\% \pm 0.186\%$, while the conventional relative L^2 error is $0.285\% \pm 0.176\%$. Thus, PIWNO attains the lower numerical error, while both methods remain in the sub-percent mean-error regime. Since the two studies use different initial-condition constructions and training/test sample counts, this comparison should be interpreted as indicative rather than strictly matched.

  The released PIWNO Burgers implementation requires approximately 46 s per epoch in our same-platform CPU reproduction. Its released script uses only 10 epochs, whereas the 300–400-epoch regime described in the PIWNO study would correspond to approximately 3.9–5.2 h at the observed rate. In comparison, the complete measured KAPI-ONet pipeline requires 36.5 min on the same CPU platform, illustrating a favorable accuracy–computational-cost tradeoff.

E.3.2 CASE II: GRF INITIAL CONDITIONS

- **GSFAE.** We consider $M_f \in \{4, 8, 16, 32, 64\}$ Gaussian splats for the GRF initial-condition representation. As shown in the left panel of Figure E.10, the validation reconstruction error decreases from 5.79% for $M_f = 4$ to 2.50% for $M_f = 8$ and 1.70% for $M_f = 16$, reaching its minimum value of 1.62% for $M_f = 32$. Increasing the representation further to $M_f = 64$ increases the validation error to 1.95%. We therefore select $M_f = 32$, corresponding to a 96-dimensional Gaussian state. The selected model gives a mean held-out initial-condition reconstruction error of 1.63%.
- **Task conditioning.** The KAPI-ONet operator is trained using 1000 initial-condition functions. The frozen 96-dimensional initial-condition state is mapped to $K = 25$ solution parents, with eight children generated per parent, giving $N = 200$ Gaussian RBFs. The conditioner contains $108\,179$ trainable parameters and is trained for $20\,000$ optimization updates. Operator training requires approximately 5803 s (96.7 min) on CPU.

  Figure E.11 shows the sampled initial conditions, their Gaussian-splat representations, task-conditioned parent and child geometries, predicted solutions, numerical references, and absolute errors for held-out cases at the 10th, 35th, 65th, and 90th error percentiles. The corresponding relative L^2 errors are 1.42%, 2.07%, 3.23%, and 6.58%, respectively.

- **Results.** Over 1000 independently held-out initial conditions, KAPI-ONet achieves a mean relative L^2 error of 3.51%, a median error of 2.53%, and a 90th-percentile error of 6.59%. The maximum observed error is 37.62%. The complete error distribution is shown in the right panel of Figure E.10. The five-model GSFAE capacity sweep requires approximately 780 s (13.0 min); together with the 5803 s (96.7 min) operator-training time, the complete measured training pipeline requires approximately 6583 s (1.83 h) on CPU.

- **Benchmarking.** To obtain a directly aligned comparison with PI-DeepONet, the KAPI-ONet experiment uses 1000 operator-training initial conditions and 1000 held-out test initial conditions, matching the training and test-set sizes used in the PI-DeepONet Burgers benchmark.

  Section 4.3 of the PI-DeepONet (Wang et al., 2021b) study reports a marked dependence of the Burgers accuracy on both the weighting of the initial-condition loss and the network architecture. Using the standard MLP architecture, the unweighted configuration ($\lambda = 1$) gives an average relative L^2 error of approximately 6%. The authors subsequently sweep $\lambda \in \{1, 5, 10, 20, 50, 100\}$, reducing the minimum error of the standard MLP to 2.80%. Replacing the standard MLP with the modified fully connected architecture of Wang et al. (2021a) further reduces the best reported error to 1.38%, attained for $\lambda = 20$.

  KAPI-ONet achieves a mean relative L^2 error of 3.51%. Thus, its error is below that of the untuned standard-MLP PI-DeepONet configuration and is close to the 2.80% minimum obtained by the standard PI-DeepONet architecture after loss-weight tuning, while the modified and tuned PI-DeepONet retains the lowest reported mean error of 1.38%. KAPI-ONet uses a fixed task-conditioned architecture throughout this experiment and does not perform a sweep over the initial-condition weight or substitute an alternative network backbone to optimize the reported Burgers accuracy.

  The difference in computational burden is also substantial. KAPI-ONet requires $20\,000$ operator-training updates and $108\,179$ trainable conditioner parameters, with operator training taking approximately 5803 s (1.61 h) on CPU. Including the complete five-model GSFAE capacity study, the full measured KAPI-ONet training pipeline requires approximately 6583 s (1.83 h) on CPU. By comparison, PI-DeepONet uses $200\,000$ optimization iterations. The authors' released Burgers implementation uses 2500 PDE-residual collocation points per training initial condition and a batch size of $50\,000$ for the initial-condition, boundary-condition, and residual data streams. The released code explicitly notes that training may encounter out-of-memory issues on Google Colab and recommends training the model on a server. This highlights the considerably larger computational and memory requirements of the PI-DeepONet Burgers configuration. Since a same-hardware wall-clock reproduction of the full PI-DeepONet Burgers training is not available here, we do not report a hardware-normalized speedup for this case.

  KAPI-ONet is trained without paired numerical solution fields, with the numerical Burgers solutions used only for held-out evaluation after model selection. Each predicted solution is represented explicitly by 200 physical space–time Gaussian RBFs. Consequently, although the modified and loss-weight-tuned PI-DeepONet attains the lower absolute solution error, KAPI-ONet provides a substantially lighter physics-informed operator training procedure, requiring one tenth as many optimization updates and producing an explicit kernel-based representation of the predicted solution.

E.4 2D STEADY INCOMPRESSIBLE NAVIER–STOKES EQUATIONS

The cavity geometry, governing equations, boundary conditions, pressure gauge, and parameter-to-field map are defined in Section D. The present subsection specifies the KAPI-ONet representation, physics-informed training, and independent FEM evaluation.

- **Task-conditioned solution representation.** No GSFAE representation is required in this example because the task input is the scalar Reynolds number rather than a spatial field. The logarithm of Re is normalized and supplied directly to the conditioner.

For each Reynolds number, KAPI-ONet generates a task-dependent Gaussian RBF representation of the three solution fields,

$$\mathbf{q}(x,y;Re)=\begin{bmatrix}u(x,y;Re)\\ v(x,y;Re)\\ p(x,y;Re)\end{bmatrix}=\mathbf{b}(Re)+\sum_{j=1}^{N}\mathbf{a}_j(Re)\varphi_j(x,y;Re),$$

where

$$\mathbf{a}_j=\begin{bmatrix}a_j^u\\ a_j^v\\ a_j^p\end{bmatrix},$$

and each anisotropic Gaussian basis function is written as

$$\varphi_j(x,y;Re)=\exp\left[-\frac{1}{2}\left(\frac{\xi_j^2}{h_{1,j}^2}+\frac{\eta_j^2}{h_{2,j}^2}\right)\right].$$

Here, (ξ_j,η_j) denote coordinates relative to the RBF center after rotation by the task-dependent orientation angle.

The representation uses $K=50$ shared reference parents and $M=8$ children per parent, giving $N=400$ physical RBFs. For each task, the conditioner predicts Reynolds-dependent shifts in the parent centers, scales, and orientations together with independent RBF coefficients for u, v, and p. The conditioner outputs $3N+5K+3=1453$ task-specific quantities and contains $263\,201$ trainable parameters.

Figure E.12 illustrates the resulting kernel geometry at low, intermediate, and high Reynolds numbers. Although the task input is only a scalar, changing Re produces coordinated changes in the parent geometry and corresponding physical child RBFs.

- **Physics-informed objective.** Spatial derivatives required by the governing equations are evaluated from the differentiable Gaussian representation. Defining the continuity and momentum residuals as

$$r_c=u_x+v_y,$$

$$r_u=u\,u_x+v\,u_y+p_x-\frac{1}{Re}\left(u_{xx}+u_{yy}\right),\qquad r_v=u\,v_x+v\,v_y+p_y-\frac{1}{Re}\left(v_{xx}+v_{yy}\right),$$

the governing-equation loss is

$$\mathcal{L}_{\mathrm{PDE}}=\left\langle r_c^2+r_u^2+r_v^2\right\rangle_{\Omega}.$$

The velocity boundary-condition loss and pressure-gauge loss are

$$\mathcal{L}_{\mathrm{BC}}=\left\langle (u-u_b)^2+(v-v_b)^2\right\rangle_{\partial\Omega},\qquad \mathcal{L}_{\mathrm{gauge}}=\left[\int_{\Omega}p(x,y)\,d\Omega\right]^2.$$

The complete objective is

$$\mathcal{L}=\mathcal{L}_{\mathrm{PDE}}+50\,\mathcal{L}_{\mathrm{BC}}+\mathcal{L}_{\mathrm{gauge}}+\mathcal{L}_{\mathrm{reg}},$$

where $\mathcal{L}_{\mathrm{reg}}$ contains weak penalties on excessive task-dependent parent shifts and RBF coefficients.

Training uses a fixed spatial collocation set containing 2209 interior PDE points and 192 boundary points, together with a 15×15 quadrature grid for the pressure gauge. Rather than using a finite corpus of precomputed tasks, Reynolds numbers are sampled uniformly from $[10,100]$ during physics-informed training.

The model is optimized for $10\,000$ updates using Adam with an initial learning rate of 10^{-3}. The learning rate is reduced by a factor of 0.5 at updates 6000 and 8500, and the gradient norm is clipped at 20. Model selection uses a deterministic physics-only validation objective evaluated at $Re=\{10,25,55,80,100\}$. The selected checkpoint occurs at update 9250, and the complete training requires approximately 8027 s (2.23 h) on CPU. No FEM velocity or pressure fields are used during training or checkpoint selection.

- **Numerical reference and results.** Independent numerical reference solutions are generated using DOLFINx with the inf–sup stable Taylor–Hood discretization $\mathbf{u}_h \in [P_2]^2$ and $p_h \in P_1$ on a 128×128 triangular cavity mesh. Reference fields are evaluated at $Re = \{10, 25, 40, 55, 70, 85, 100\}$ and are used only after KAPI-ONet training.

  Figure E.13 compares the predicted velocity magnitude with the FEM reference for representative low, intermediate, and high Reynolds numbers. Across the seven reference cases, KAPI-ONet achieves a mean relative L^2 error of 15.55% in the velocity magnitude and 20.97% in the complete velocity vector. The discrepancy increases moderately with Reynolds number, with the largest local differences concentrated near the moving lid and upper recirculation region.

  Classical lid-driven-cavity centerline diagnostics are shown in Figure E.14. The vertical-centerline horizontal velocity $u(0.5, y)$ has a mean relative L^2 error of approximately 9.75% over the seven reference Reynolds numbers, while the corresponding mean error for the horizontal-centerline vertical velocity $v(x, 0.5)$ is approximately 16.15%. The vertical-centerline profile remains closely aligned with the numerical reference across the parameter range, whereas the horizontal-centerline discrepancy becomes more pronounced toward the upper end of the Reynolds-number interval.

- **Interpretation.** This experiment serves as a bridge between operator learning and parametric learning. In the preceding PDE examples, the task is specified by a spatially varying input function and may therefore benefit from an explicit input representation such as GS-FAE. Here, the task is specified only by the scalar Reynolds number, and no functional input encoder is required. Nevertheless, the output retains the same task-conditioned parametric kernel representation: changes in Re generate coordinated changes in the RBF geometry and coefficients defining the continuous solution fields.

  The example therefore illustrates that the defining component of KAPI-ONet is not a particular representation of the input, but the generation of an explicit task-conditioned kernel representation of the output. The same framework can consequently accommodate both function-to-field operator learning and finite-dimensional parameter-to-field mappings. Since no directly matched operator-learning benchmark is used for this experiment, the independent FEM solutions provide the numerical reference for assessing the learned parametric solution family.

E.5 Sparse-observation inverse hemodynamics and wall shear stress

The stenotic geometry, governing equations, known and hidden boundary conditions, sensor locations, and inverse operator are defined in Section D. The dataset contains 200 training, 50 validation, and 50 independently held-out test realizations; full FEM fields are used only for post-training evaluation.

- **Sparse observations.** For each realization, the only measured quantities supplied to the conditioner are the two velocity components at five fixed interior locations,

$$(0.50, 0.000), \quad (0.58, 0.016), \quad (0.66, -0.016),$$
$$(0.74, 0.012), \quad (0.82, -0.012).$$

  The operator input is therefore

$$\mathbf{s} = [u_1, v_1, \ldots, u_5, v_5] \in \mathbb{R}^{10},$$

  and the inverse mapping of interest is

$$\mathcal{G}_{\text{inv}} : \mathbf{s} \mapsto \big(u(\cdot,\cdot), v(\cdot,\cdot), p(\cdot,\cdot)\big).$$

- **Task-conditioned solution representation.** KAPI-ONet represents the solution through a task-conditioned streamfunction–pressure RBF model. The conditioner maps the 10-dimensional sensor vector to $K = 50$ solution parents, with eight children generated per parent, giving $N = 400$ anisotropic Gaussian RBFs.

  Let

$$\widetilde{\psi}(x, y; \mathbf{s}) = \sum_{j=1}^{N} a_j^{\psi}(\mathbf{s}) \varphi_j(x, y; \mathbf{s}),$$

and

$$\widetilde{p}(x, y; \mathbf{s}) = \sum_{j=1}^{N} a_j^p(\mathbf{s})\varphi_j(x, y; \mathbf{s}),$$

where the RBF centers, anisotropic scales, orientations, and coefficients are generated conditionally from $\mathbf{s}$. Introducing the normalized cross-channel coordinate

$$\eta(x, y) = \frac{y + h(x)}{2h(x)},$$

the streamfunction is parameterized as

$$\psi(x, y; \mathbf{s}) = Q_0 q(\mathbf{s}) F(\eta) + Q_0 B(\eta) \left[b_\psi(\mathbf{s}) + \widetilde{\psi}(x, y; \mathbf{s}) \right],$$

where

$$F(\eta) = 3\eta^2 - 2\eta^3, \qquad B(\eta) = \eta^2(1 - \eta)^2.$$

The pressure is represented as

$$p(x, y; \mathbf{s}) = P_0 \widetilde{p}(x, y; \mathbf{s}),$$

with Q_0 and P_0 fixed reference scales.

Velocity is obtained directly from the streamfunction,

$$u = \frac{\partial \psi}{\partial y}, \qquad v = -\frac{\partial \psi}{\partial x}.$$

Consequently, $\nabla \cdot \mathbf{u} = 0$ holds identically. The functions F and B also enforce the no-slip wall behavior through the ansatz, so neither a continuity-loss term nor a separate wall-boundary penalty is required. The conditioner contains $279\,510$ trainable parameters.

Figure E.15 illustrates the complete mapping for held-out cases selected at the 10th, 35th, 65th, and 90th percentiles of the velocity-error distribution. The rows show the five sparse velocity measurements, task-conditioned parent and child geometries, KAPI-ONet and FEM velocity magnitudes, velocity error, and the corresponding pressure fields.

- **Physics-informed inverse objective.** Since incompressibility and the no-slip wall behavior are incorporated into the ansatz, training is driven by the two momentum equations together with the sparse measurements. Defining

$$r_u = \rho \left(u\, u_x + v\, u_y \right) - \mu \left(u_{xx} + u_{yy} \right) + p_x,$$

$$r_v = \rho \left(u\, v_x + v\, v_y \right) - \mu \left(v_{xx} + v_{yy} \right) + p_y,$$

the momentum loss is

$$\mathcal{L}_{\mathrm{mom}} = \left\langle r_u^2 + r_v^2 \right\rangle_\Omega .$$

Agreement with the measured velocity components is enforced through

$$\mathcal{L}_{\mathrm{sens}} = \frac{1}{5U_{\mathrm{ref}}^2} \sum_{i=1}^{5} \left[\big(u(\mathbf{x}_i) - u_i \big)^2 + \big(v(\mathbf{x}_i) - v_i \big)^2 \right].$$

The complete objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{mom}} + \mathcal{L}_{\mathrm{sens}} + \mathcal{L}_{\mathrm{reg}},$$

where $\mathcal{L}_{\mathrm{reg}}$ contains weak penalties on excessive parent-geometry shifts, RBF coefficients, and child centers leaving the computational domain. No full-field FEM supervision is used.

The model is trained for $10\,000$ updates, with the checkpoint selected using the physics-plus-sensor validation objective. The selected checkpoint occurs at update 9000. Training requires approximately 9006 s (2.50 h) on CPU.

- **Wall shear stress and results.** Wall shear stress is evaluated from the reconstructed velocity gradients. For unit tangent $\mathbf{t}$ and outward normal $\mathbf{n}$,

$$\tau_w = \mu\, \mathbf{t}^T \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T \right) \mathbf{n}.$$

For KAPI-ONet, the required gradients are obtained directly from the continuous differentiable RBF representation. The FEM references store velocity values rather than wall gradients; reference WSS is therefore reconstructed independently using a no-slip-constrained one-sided quadratic fit to the near-wall FEM velocity field. WSS is evaluated on both walls over $x \in [0.40, 0.95]$.

Across the 50 held-out problems, the mean relative L^2 errors of the reconstructed velocity and pressure fields are 21.51% and 27.60%, respectively. The hidden inlet profile is considerably less identifiable, with a mean relative L^2 error of 84.43%. Nevertheless, the corresponding flow rate is recovered with a mean relative error of 4.30%, while the combined upper- and lower-wall WSS has a mean relative L^2 error of 4.51% and a median error of 3.02%.

The contrast is shown directly in Figure E.16. Although the inferred inlet profiles can differ substantially from the hidden FEM profiles, the WSS curves in the stenotic and downstream region closely follow the FEM references.

- **Interpretation.** This experiment differs fundamentally from the preceding forward operator-learning benchmarks. The inverse problem is underdetermined: five interior velocity measurements do not uniquely determine the hidden inlet profile, and different upstream conditions may produce similar downstream flow behavior. Exact recovery of the inlet boundary condition is therefore not required for successful recovery of the clinically relevant quantity of interest.

  The results show that KAPI-ONet can reconstruct the downstream flow sufficiently well to recover WSS with approximately 4.5% mean error even though the hidden inlet profile itself remains poorly identified. This behavior is consistent with the inverse-hemodynamics observation of Arzani et al. (2021), but here the repeated inverse solve is amortized into a single task-conditioned operator over a family of flows. No directly matched operator-learning benchmark is used for this setting; the hidden FEM solutions serve only as post-training references for assessing the reconstructed fields and wall shear stress.

E.6 Weakly supervised variational segmentation

- **Data split and task conditioning.** The operator-learning formulation is defined in Section D. We use 100 subjects from the ACDC dataset (Bernard et al., 2018) at end diastole and construct a subject-level split of $70/15/15$ subjects for training, validation, and testing, corresponding to $350/75/75$ image slices. For each image, the available weak annotation is a single point $c_0 \in \mathbb{R}^2$ located inside the LV cavity.

  Unlike the PDE examples, GSFAE is not used in this problem. GSFAE is an optional input-representation mechanism rather than an intrinsic component of KAPI-ONet. A global Gaussian reconstruction of the MR image would allocate representation capacity to multiple anatomical structures and could smooth local intensity transitions that are informative for contour localization. Instead, the task conditioner receives the normalized MR image directly, together with a Gaussian heatmap centered at the weak localization point.

  For each task $\mathcal{T} = (I, c_0)$, KAPI-ONet generates the parameters of a continuous implicit RBF field,

$$\phi_\Theta(x, y) = b + \sum_{k=1}^{K} a_k \exp\left(-\frac{\|(x, y) - \mathbf{c}_k\|_2^2}{2h_k^2}\right), \qquad K = 25,$$

  where $\mathbf{c}_k$, h_k, and a_k denote the RBF centers, widths, and coefficients, respectively. The LV annotation is represented by the zero level set,

$$\partial\Omega_{\mathrm{LV}} = \{(x, y) : \phi_\Theta(x, y) = 0\}, \qquad \Omega_{\mathrm{LV}} = \{(x, y) : \phi_\Theta(x, y) < 0\}.$$

  The resulting task state contains 101 parameters: 50 center coordinates, 25 widths, 25 coefficients, and one bias. The task-conditioned generator contains $114\,080$ trainable parameters. Training uses 350 weakly annotated images and requires approximately 3.1 min on CPU; dense expert masks are not used in generator training.

- **Weakly supervised variational objective.** Unlike the preceding PDE examples, the desired contour is not determined by a fundamental conservation law whose residual can be

Table E.1: Loss terms used for weakly supervised cardiac annotation. Here $S = H_\tau(-\phi)$ denotes the soft predicted interior, $B = \delta_\tau(\phi)\|\nabla\phi\|$ the soft boundary density, $\mathcal{R}$ the local search region, and c_0 the weak localization point.

TERM	EXPRESSION	ROLE
$\mathcal{L}_{\mathrm{CV}}$	$\frac{1}{\lvert\mathcal{R}\rvert}\int_{\mathcal{R}}\left[S(I-c_{\mathrm{in}})^2+(1-S)(I-c_{\mathrm{out}})^2\right]d\mathbf{x}$	region fitting
$\mathcal{L}_{\mathrm{geo}}$	$\langle g(I)\,B\rangle$	attraction to image edges
$\mathcal{L}_{\mathrm{per}}$	$\langle B\rangle$	contour regularity
$\mathcal{L}_{\mathrm{area}}$	$\left[\max\left(\left\lvert\frac{A}{A_\star}-1\right\rvert-\epsilon_A,\,0\right)\right]^2$	plausible region size
$\mathcal{L}_{\mathrm{centroid}}$	$\frac{\lVert\bar{\mathbf{c}}-c_0\rVert_2^2}{r_0^2}$	weak-point localization
$\mathcal{L}_{\mathrm{leak}}$	$\frac{\int(1-\mathbf{1}_{\mathcal{R}})S\,d\mathbf{x}}{\int S\,d\mathbf{x}}$	suppress distant foreground
$\mathcal{L}_{\mathrm{RBF}}$	$\lambda_{\mathbf{c}}\frac{\langle\lVert\mathbf{c}-\mathbf{c}^0\rVert_2^2\rangle}{r_0^2}+\lambda_h\left\langle(\log h-\log h^0)^2\right\rangle+\lambda_a\left\langle(a-a^0)^2\right\rangle$	RBF-state regularization
$\mathcal{L}_{\mathrm{shape}}$	$1-\mathrm{Dice}_{\mathrm{soft}}(S,S_0)$	correction trust region
$\mathcal{L}_{\text{area-trust}}$	$\left[\log\left(\frac{A}{A_0}\right)\right]^2$	preserve generated area

minimized directly. Instead, the supervision is provided by an image-based variational model. Because cardiac MR images contain noise, nonuniform contrast, and multiple anatomical structures with overlapping intensity ranges, we organize the objective into three functional groups,

$$\mathcal{L}_{\mathrm{var}} = \underbrace{\mathcal{L}_{\mathrm{image}}}_{\text{image-driven contour}} + \underbrace{\mathcal{L}_{\mathrm{loc}}}_{\text{weak localization}} + \underbrace{\mathcal{L}_{\mathrm{RBF}}}_{\text{representation regularization}}.$$

The image-driven component is

$$\mathcal{L}_{\mathrm{image}} = w_{\mathrm{CV}}\mathcal{L}_{\mathrm{CV}} + \lambda_g\mathcal{L}_{\mathrm{geo}} + \lambda_p\mathcal{L}_{\mathrm{per}},$$

where $\mathcal{L}_{\mathrm{CV}}$ is the principal Chan–Vese region-fitting energy (Chan & Vese, 2001). Defining $H_\tau(-\phi)$ as a smooth approximation of the predicted interior region, the localized Chan–Vese term is

$$\mathcal{L}_{\mathrm{CV}} = \frac{1}{\lvert\mathcal{R}\rvert}\int_{\mathcal{R}}\left[H_\tau(-\phi)(I-c_{\mathrm{in}})^2+\big(1-H_\tau(-\phi)\big)(I-c_{\mathrm{out}})^2\right]d\mathbf{x},$$

where $\mathcal{R}$ is a search region centered at c_0, and c_{in} and c_{out} are the corresponding soft inside/outside mean intensities. The geodesic term weights the zero-level-set length by an edge indicator obtained from image gradients and therefore encourages the contour toward local boundaries, while the perimeter term regularizes excessive contour irregularity.

The weak-localization component,

$$\mathcal{L}_{\mathrm{loc}} = \lambda_A\mathcal{L}_{\mathrm{area}} + \lambda_c\mathcal{L}_{\mathrm{centroid}} + \lambda_\ell\mathcal{L}_{\mathrm{leak}},$$

addresses ambiguities that cannot be resolved from image intensity alone. The centroid term keeps the predicted region near the clinician-provided point, the leakage term penalizes predicted foreground outside the local search region, and the area term prevents implausible collapse or excessive expansion while allowing substantial variation in LV size.

Finally, $\mathcal{L}_{\mathrm{RBF}}$ regularizes changes in RBF centers, widths, and coefficients and keeps the continuous parametric representation well behaved. The RBF parameters are additionally constrained to admissible spatial and scale ranges. Thus, the auxiliary terms do not provide dense segmentation supervision; rather, they encode weak geometric admissibility needed to apply a variational contour model to heterogeneous real images. The full description of loss terms is given in Table E.1.

Table E.2: Held-out cardiac annotation accuracy over 75 test images. Expert masks are not used during KAPI-ONet prediction or variational correction and are used here only to compute the reported test Dice.

METHOD	DICE
KAPI-ONet prediction	0.868 ± 0.068
KAPI-ONet + variational correction	$\mathbf{0.912 \pm 0.062}$

- **Mask-free variational correction.** The frozen KAPI-ONet generator first produces an initial field ϕ_{Θ_0} for each unseen image. We subsequently allow a lightweight per-image correction of the 101-dimensional RBF state while keeping the generator fixed. The correction minimizes the same image-based variational objective together with a trust term,

$$\mathcal{L}_{\text{ref}} = \mathcal{L}_{\text{var}} + \lambda_s \mathcal{L}_{\text{shape}} + \lambda_t \mathcal{L}_{\text{area-trust}},$$

  where $\mathcal{L}_{\text{shape}}$ is a soft-Dice distance between the corrected region and the original KAPI-ONet prediction, and

$$\mathcal{L}_{\text{area-trust}} = \left[\log\left(\frac{A}{A_0}\right)\right]^2$$

  penalizes excessive expansion or contraction relative to the generated region. These terms constrain the correction to remain a local adaptation of the operator prediction rather than an independent contour optimization.

  The correction uses 250 optimization steps with fixed settings for every image. The Chan–Vese weight is selected on the validation set by sweeping $w_{\text{CV}} \in \{1, \ldots, 10\}$; $w_{\text{CV}} = 2$ gives the highest mean validation Dice and is fixed before test evaluation. Validation masks are used only for this hyperparameter selection. The correction itself and its safeguard remain mask-free. A corrected field is rejected in favor of the original KAPI-ONet prediction if its overlap with the generated region falls below 0.70, if its area changes by a factor outside $[0.70, 1.40]$, or if the trust-region objective does not improve.

  Figure E.17 illustrates the complete representation for representative held-out cases. The first row shows the input MRI and weak localization point; the second row shows the generated task-dependent RBF geometry and its zero level set; and the final row shows the corrected annotation together with the expert reference. The examples are selected according to quantiles of the KAPI-ONet-generated region area, rather than using ground-truth accuracy for visual selection.

- **Results.** On the 75 images from 15 independently held-out subjects, the frozen KAPI-ONet prediction achieves a mean Dice score of 0.868 ± 0.068. The fixed variational correction increases the mean Dice to 0.912 ± 0.062, as summarized in Table E.2. The correction improves 82.7% of the held-out cases. Figure E.18 compares the generated and corrected Dice scores image by image. The frozen operator requires approximately 0.041 s per image on CPU, while the optional variational correction requires approximately 4.2 s per image. Although performance could be enhanced further with LV-specific anatomical priors (e.g., convexity constraints (Shi & Li, 2021)), we prioritized demonstrating the core methodological framework over max-score tuning.

- **Interpretation.** This experiment is intended as weakly supervised image annotation rather than as a direct comparison with fully supervised segmentation networks such as U-Net (Azad et al., 2024). Dense expert masks are not supplied to the KAPI-ONet generator and do not enter the per-image variational correction. Instead, the model learns to map an MR image and a minimal localization cue to an explicit continuous RBF geometry whose zero level set provides a candidate LV annotation. The expert masks are used for validation-based selection of the single correction hyperparameter and for final held-out evaluation.

  The attainable agreement is therefore governed by the information contained in the weak variational model rather than by direct supervision from the desired contour. In particular, the Chan–Vese energy is a mathematical model of image-region homogeneity rather than a fundamental physical law, and cardiac MR images exhibit substantial variability in contrast,

noise, image quality, and surrounding structures with similar intensity. Exact recovery of the expert contour is therefore not encoded by the training objective.

The purpose of the experiment is to test whether a per-image variational annotation problem can be amortized into a reusable task-conditioned operator. KAPI-ONet provides an immediate continuous annotation from a single weak cue, after which a short mask-free variational correction can adapt the explicit RBF geometry to the individual image. The intended use is to assist clinicians in generating or refining annotation masks that may subsequently be used as labeled data for conventional supervised segmentation models.

F ARCHITECTURE DETAILS

F.1 TASK REPRESENTATION AND OUTPUT PARAMETERIZATION

KAPI-ONet separates the representation of the task input from the explicit parametric representation of the output field. For functional PDE inputs, GSFAE is used when a compact Gaussian representation is advantageous. The lid-driven cavity instead conditions directly on Re; the inverse-hemodynamics case conditions on the 10-dimensional sensor vector; and the cardiac annotation model conditions directly on the normalized image together with the localization heatmap. In all cases, the conditioner produces a task-specific continuous kernel field rather than a grid of output values.

For the PDE operator benchmarks with GSFAE, the solution representation uses $K = 25$ parents and eight children per parent, giving $N = 200$ physical Gaussian RBFs. The lid-driven cavity and inverse-hemodynamics cases use $K = 50$ parents and eight children per parent, giving $N = 400$ RBFs. The cardiac annotation model directly generates $K = 25$ RBFs whose zero level set defines the LV boundary.

F.2 GSFAE ARCHITECTURE

GSFAE is used only for the five experiments in which the task input is a spatial function: the two Poisson cases, diffusion–reaction, and the two Burgers cases. It consists of a neural encoder that predicts the parameters of a Gaussian representation together with an analytic Gaussian decoder; there is no trainable neural decoder. After GSFAE training, the selected encoder is frozen and is used only to generate the task state supplied to KAPI-ONet.

For the two-dimensional Poisson inputs, the encoder receives the sampled forcing field as a single-channel image. Four strided convolutional layers are used with channel widths 32, 64, 128, and 128. The first convolution uses a 5×5 kernel and the remaining three use 3×3 kernels; all four use stride two and GELU activations. The convolutional features are reduced by 2×2 adaptive average pooling and mapped through a $512 \to 256$ fully connected layer followed by GELU and LayerNorm. Three linear heads then predict the Gaussian amplitudes, two-dimensional centers, and isotropic widths. Hence, each input splat contributes four state variables and the GSFAE state has dimension $4M_f$.

For the one-dimensional diffusion–reaction and Burgers inputs, the encoder is a fully connected network with three hidden layers of 256 neurons. Each hidden layer uses GELU followed by LayerNorm. Three independent linear heads predict the amplitude, center, and width of each one-dimensional Gaussian splat, giving a state dimension of $3M_f$.

The experiment-specific input and selected GSFAE capacities are summarized in Table F.1.

For the PI-DeepONet-matched diffusion–reaction experiment, the same one-dimensional GSFAE architecture is used with $M_f = 64$, giving a 192-dimensional forcing state.

F.3 TASK-CONDITIONER ARCHITECTURE

For the first five PDE experiments, the frozen Gaussian task state is standardized and supplied to a fully connected task conditioner. The conditioner has three hidden layers, each containing 149 neurons with Tanh activation. Its output dimension is $N + 5K + 1 = 326$: $N = 200$ solution

Table F.1: GSFAE architectures used for functional task inputs. The Gaussian state is frozen before KAPI-ONet operator training.

PROBLEM	INPUT	ENCODER	ACTIVATION / NORM.	M_f	STATE DIM.
Poisson, localized Gaussian	64×64	4 Conv + FC-256	GELU; final LayerNorm	4	16
Poisson, sinusoidal	65×65	4 Conv + FC-256	GELU; final LayerNorm	64	256
Diffusion–reaction	100 samples	3 FC $\times$ 256	GELU + LayerNorm	32	96
Burgers, parametric IC	100 samples	3 FC $\times$ 256	GELU + LayerNorm	32	96
Burgers, GRF IC	101 samples	3 FC $\times$ 256	GELU + LayerNorm	32	96

coefficients, five task-dependent geometric shifts for each of the $K = 25$ parents, and one solution bias. Only the input dimension changes with the selected GSFAE capacity.

The lid-driven cavity retains the same three-layer, 149-neuron Tanh conditioner, but the input is the normalized logarithmic Reynolds number. Its output dimension is $3N + 5K + 3 = 1453$, corresponding to three independent coefficient blocks for (u, v, p), five geometric shifts for each of the $K = 50$ parents, and three field biases.

For inverse hemodynamics, the standardized 10-dimensional sensor vector is mapped through three hidden layers of 192 neurons with Tanh activations. The output dimension is $2N + 5K + 2 = 1052$: two $N = 400$ coefficient blocks for the streamfunction correction and pressure, five geometric shifts for each of the $K = 50$ parents, a flow-rate factor, and a streamfunction bias.

The cardiac annotation conditioner uses a different architecture because the task contains an image. The normalized MRI and localization heatmap form a two-channel 112×112 input. The image encoder contains four downsampling convolutional blocks with channel progression $2 \to 8 \to 16 \to 32 \to 64$. Each block contains two 3×3 convolutions with GroupNorm and SiLU activation, with the first convolution of each block using stride two. Global adaptive average pooling produces a 64-dimensional image feature. The normalized localization coordinate is encoded separately by a $2 \to 16 \to 16$ SiLU MLP. The two representations are concatenated into an 80-dimensional task feature and passed through the hypernetwork $80 \to 128 \to 128 \to 101$, with SiLU activations in the two hidden layers. The 101 outputs correspond to $2K$ center shifts, K width shifts, K coefficient shifts, and one bias for $K = 25$ direct RBFs.

Table F.2 summarizes the resulting conditioner architectures. The reported parameter count is the total number of trainable KAPI-ONet parameters for that experiment, including the neural conditioner and the shared trainable output-representation parameters described below; frozen GSFAE parameters are excluded.

F.4 Solution architecture and parameterization

For the scalar PDE experiments, the continuous solution has the common form

$$\widehat{u}(\mathbf{x}; \mathcal{T}) = b(\mathcal{T}) + \sum_{n=1}^{N} a_n(\mathcal{T}) \varphi_n(\mathbf{x}; \mathcal{T}),$$

where $\mathcal{T}$ denotes the task. The N physical Gaussian RBFs are organized hierarchically using K parents and M children per parent, with $N = KM$.

Each parent possesses a center, two anisotropic scales, and an orientation. The reference parent centers, scales, and orientations are shared across all tasks and are globally trainable. For a given task, the conditioner predicts five bounded parent-geometry shifts per parent: two center shifts, two scale shifts, and one orientation shift. The resulting task-conditioned parent geometry is therefore obtained by perturbing a shared trainable reference geometry rather than predicting every physical RBF independently.

Table F.2: Task-conditioner architectures. The parameter count includes the neural conditioner and the shared trainable parameters of the corresponding output representation; frozen GSFAE parameters are excluded.

PROBLEM	INPUT	CONDITIONER	ACT.	PARAMETERS
Poisson, localized Gaussian	16-D GSFAE	$16-149-149-149-326$	Tanh	96 259
Poisson, sinusoidal	256-D GSFAE	$256-149-149-149-326$	Tanh	132 019
Diffusion–reaction	96-D GSFAE	$96-149-149-149-326$	Tanh	108 179
Diffusion–reaction, matched	192-D GSFAE	$192-149-149-149-326$	Tanh	122 483
Burgers, parametric IC	96-D GSFAE	$96-149-149-149-326$	Tanh	108 179
Burgers, GRF IC	96-D GSFAE	$96-149-149-149-326$	Tanh	108 179
Lid-driven cavity	normalized $\log Re$	$1-149-149-149-1453$	Tanh	263 201
Inverse hemodynamics	10-D sensor vector	$10-192-192-192-1052$	Tanh	279 510
Cardiac annotation	MRI + localization cue	CNN + $80-128-128-101$	SiLU	114 080

The child construction is non-trainable. Each parent is associated with fixed latent child offsets and fixed child-width multipliers generated once before training and stored as model buffers. The physical child centers, widths, and orientations are then derived deterministically from the task-conditioned parent center, anisotropic scales, and orientation. Consequently, the N physical child RBFs do not introduce independent trainable geometric parameters. The conditioner directly predicts only their field coefficients, while the child geometry is inherited from the parent representation.

For the first five scalar PDE cases, $K = 25$ parents and $M = 8$ children give $N = 200$ physical anisotropic RBFs. A single task-conditioned coefficient block and one bias define the scalar solution field. The model additionally contains one shared trainable coefficient scale.

The lid-driven cavity uses the same hierarchical construction with $K = 50$, $M = 8$, and $N = 400$. A common task-conditioned RBF geometry is shared by the three primary variables, while independent coefficient blocks and biases represent u, v, and p:

$$\mathbf{q}(x, y; Re) = \mathbf{b}(Re) + \sum_{n=1}^{400} \mathbf{a}_n(Re)\, \varphi_n(x, y; Re), \qquad \mathbf{q} = \begin{bmatrix} u \\ v \\ p \end{bmatrix}.$$

The model contains three shared trainable coefficient scales, one for each field component.

The inverse-hemodynamics model also uses $K = 50$, $M = 8$, and $N = 400$ physical anisotropic RBFs, but the represented variables are a raw streamfunction correction and pressure. The same task-conditioned RBF geometry is used for both coefficient blocks. In addition to the two coefficient blocks and the five parent shifts, the conditioner predicts a flow-rate factor and a streamfunction bias. The physical velocity is not represented by an independent RBF coefficient block; instead it is derived from the streamfunction through

$$u = \frac{\partial \psi}{\partial y}, \qquad v = -\frac{\partial \psi}{\partial x},$$

which makes the velocity field divergence-free by construction. The latent child offsets and width multipliers remain fixed, and all child geometry is derived from the task-conditioned parents.

The cardiac annotation model does not use the parent–child hierarchy. It directly represents the implicit field using $K = 25$ isotropic Gaussian RBFs,

$$\phi(x, y) = b + \sum_{k=1}^{25} a_k \exp\left(-\frac{\|(x, y) - \mathbf{c}_k\|_2^2}{2h_k^2} \right).$$

The shared trainable quantities are the 25 reference offsets relative to the localization point and the 25 base log-widths. For each image, the conditioner predicts 50 center displacements, 25 log-width

Table F.3: Task-conditioned output architectures. Children are generated deterministically from their task-conditioned parents and do not introduce independent geometric parameters. The cardiac model directly generates its RBFs and therefore has no parent–child hierarchy.

PROBLEM	FIELDS	PARENTS / RBFs	CHILDREN	TOTAL RBFs	OUTPUT DIMENSION
Scalar PDEs	scalar u or s	25 parents	8 / parent	200	$N + 5K + 1 = 326$
Lid-driven cavity	(u, v, p)	50 parents	8 / parent	400	$3N + 5K + 3 = 1453$
Inverse hemodynamics	(ψ, p)	50 parents	8 / parent	400	$2N + 5K + 2 = 1052$
Cardiac annotation	ϕ	25 direct RBFs	–	25	$4K + 1 = 101$

shifts, 25 coefficient shifts, and one bias. Thus, unlike the PDE architectures, all 25 RBFs are generated directly and no child geometry is derived from parents. The zero level set of the resulting field defines the cardiac annotation.

The distinction between shared trainable, task-conditioned, and derived quantities is summarized in Table F.3.

G Out-of-distribution performance evaluation

G.1 Out-of-distribution correlation-length shift

We next test distribution shift in the functional input of the diffusion–reaction problem. Training forcings are sampled from the same zero-mean squared-exponential GRF family with correlation length $\ell_{\text{train}} = 0.20$. At test time, the frozen GSFAE and KAPI-ONet are evaluated at $\ell \in \{0.10, 0.15, 0.20, 0.25, 0.30, 0.40\}$, using 100 forcing realizations at each length scale. The same underlying standard-normal draws are transformed by the covariance kernel at each ℓ, reducing sampling variability when comparing different shifts. No OOD data are used for retraining, fine-tuning, normalization, calibration, or checkpoint selection. We treat $\ell \in \{0.15, 0.25, 0.30\}$ as near OOD and $\ell \in \{0.10, 0.40\}$ as far OOD, while $\ell = 0.20$ is the in-distribution reference.

Figure G.1 shows both the solution error and the frozen GSFAE reconstruction error. At the training correlation length, KAPI-ONet attains a mean solution relative L_2 error of 0.69%, with a 90th percentile of 1.09%, while the mean GSFAE reconstruction error is 0.72%. Moderate shifts toward smoother inputs remain well controlled: at $\ell = 0.25$ and 0.30, the mean solution errors are 0.55% and 0.56%, respectively, and even at the far-OOD value $\ell = 0.40$ the mean error remains 0.76%. In contrast, extrapolation toward shorter correlation lengths is substantially more difficult. At $\ell = 0.15$, the mean solution error increases to 3.43%, and at $\ell = 0.10$ it reaches 21.15%, with a 90th-percentile error of 38.42%. The corresponding GSFAE error at $\ell = 0.10$ rises to 13.19%, indicating that a substantial part of the degradation is already associated with representing the rougher forcing functions outside the encoder's training distribution.

Figure G.2 gives a spatial view using task index zero from the paired Gaussian draws at $\ell = 0.10$, 0.20, and 0.40; the displayed cases are therefore fixed independently of prediction accuracy. For the in-distribution case $\ell = 0.20$, the forcing reconstruction and space–time solution are both closely reproduced, with solution and forcing errors of 0.62% and 0.67%, respectively. The smoother far-OOD case $\ell = 0.40$ remains similarly accurate, with corresponding errors of 0.55% and 0.70%. By contrast, at $\ell = 0.10$ the forcing contains substantially finer spatial variation that is not reproduced as accurately by the frozen GSFAE; the representative forcing error increases to 17.79% and the resulting solution error to 12.77%. This directional asymmetry is consistent with the aggregate results: extrapolation toward smoother forcings is substantially more robust than extrapolation toward shorter correlation lengths and higher-frequency input structure.

G.2 Out-of-distribution parameter extrapolation

We evaluate the frozen parametric Burgers operator beyond the training domain $(\zeta, \eta) \in [0.5, 1.5]^2$. No OOD samples are used for retraining, fine-tuning, normalization, calibration, or checkpoint selection. Evaluation is performed on a deterministic 17×17 grid spanning $(\zeta, \eta) \in [0, 2]^2$. We define the Euclidean distance d_{box} to the training box and classify cases with $0 < d_{\mathrm{box}} \leq 0.25$ as near OOD and those with $d_{\mathrm{box}} > 0.25$ as far OOD. This gives 81 in-distribution, 76 near-OOD, and 132 far-OOD tasks.

Figure G.3 shows the solution relative L_2 error over the complete parameter plane. Mean error is 0.33% inside the training support, increases to 10.61% for near-OOD tasks, and reaches 41.28% for far-OOD tasks. The corresponding medians are 0.26%, 3.10%, and 21.01%, respectively, indicating that the far-OOD average is influenced by a subset of substantially more difficult extrapolation directions. The error map also shows that degradation is strongly nonuniform across the parameter plane, with considerably larger errors toward the low-η, high-ζ region.

Because this case uses GSFAE to represent the functional initial condition, we also monitor the input-reconstruction error separately. Its mean relative L_2 error increases from 0.19% for in-distribution tasks to 5.56% for near-OOD and 24.69% for far-OOD tasks. Thus, part of the degradation under strong extrapolation arises before the PDE operator itself, as the initial condition moves away from the distribution on which the frozen GSFAE was trained.

Figure G.4 illustrates four parameter locations selected a priori, independently of prediction accuracy. The two near-OOD cases, $(\zeta, \eta) = (0.25, 1.00)$ and $(1.75, 1.00)$, retain accurate initial-condition representations and yield solution errors of 3.92% and 5.43%, respectively. At the farther corners, the extrapolation becomes markedly more difficult: $(0, 2)$ gives 33.92% solution error, while $(2, 0)$ gives 123.43%. The latter case also exhibits a substantial mismatch in the frozen GSFAE representation of the initial condition, consistent with the increasing input-representation error observed across the far-OOD regime.

G.3 Out-of-distribution Reynolds-number extrapolation

We further evaluate the frozen cavity operator outside the training interval $Re \in [10, 100]$. No OOD samples are used for retraining, fine-tuning, normalization, calibration, or checkpoint selection. The same logarithmic Reynolds-number conditioning used during training is retained, so extrapolation is performed directly beyond the learned task interval. We consider $Re = \{5, 7.5, 110, 125, 150\}$ as OOD cases and retain $Re = \{10, 55, 100\}$ as in-distribution anchors. Independent Taylor–Hood FEM solutions are generated with the same reference procedure used for the in-distribution cavity study and sampled on the common 201×201 evaluation grid.

Figure G.5 shows the resulting relative L_2 errors. The velocity-magnitude error changes from 14.77% at $Re = 10$ and 18.24% at $Re = 100$ to 19.15% at $Re = 5$ and 23.48% at $Re = 150$. Thus, the frozen operator exhibits a gradual degradation rather than an abrupt failure when evaluated beyond its training support. The vertical centerline $u(0.5, y)$ follows a similar trend, increasing to 18.71% at $Re = 150$. The horizontal centerline $v(x, 0.5)$ is more sensitive to upper-range extrapolation, reaching 37.12% at $Re = 150$.

The field comparisons in Fig. G.6 provide the corresponding spatial view at the two farthest extrapolation points. At $Re = 5$, KAPI-ONet closely reproduces the FEM velocity-magnitude distribution, with the largest discrepancies confined primarily to the upper portion of the cavity near the moving lid. At $Re = 150$, the dominant lid-driven high-speed layer and the overall interior speed distribution remain identifiable, while larger errors develop along the upper and right portions of the cavity and around the interior low-speed region. These spatial discrepancies are consistent with the increase in the global and centerline errors observed beyond the upper training boundary.

H Effect of parent–child factorization at fixed RBF budget

We study the effect of the parent–child factorization on the parametric viscous Burgers test case while keeping the total number of physical solution RBFs fixed. The same frozen 96-dimensional GSFAE representation of the initial condition is used for every variant. We also keep the 2000

Table H.1: Parent–child structural ablation for the parametric Burgers test case at a fixed physical RBF budget of $N = 200$. All variants use the same frozen GSFAE, task splits, hidden conditioner architecture, physics-informed training protocol, and 250 held-out test tasks.

K	M_{child}	Conditioner output dim.	Trainable params.	Mean L^2 (%)	Max L^2 (%)
10	20	251	96 854	0.302	1.065
20	10	301	104 404	0.286	0.949
25	8	326	108 179	0.270	0.921
40	5	401	119 504	0.235	1.045
50	4	451	127 054	0.228	0.604

operator-training tasks, 250 held-out test tasks, task-state normalization, conditioner hidden architecture, collocation strategy, loss weights, optimizer, learning-rate schedule, and 10 000 training updates unchanged.

The physical RBF budget is fixed at $N = 200$, while the factorization $N = KM_{\text{child}}$ is varied over $(K, M_{\text{child}}) \in \{(10, 20), (20, 10), (25, 8), (40, 5), (50, 4)\}$. Smaller K corresponds to stronger geometric sharing because more children inherit the geometry of each parent, whereas larger K provides more independently task-adaptive parent supports. Since the conditioner predicts five geometric quantities per parent, reducing K decreases both the conditioner output dimension and the total number of trainable parameters even though the number of physical basis functions remains fixed.

The purpose of this study is not to tune K for the lowest possible test error, but to examine the parameter-efficiency induced by the parent–child geometry. Throughout the main experiments, the factorization $K = 25$, $M_{\text{child}} = 8$ was used without an accuracy-oriented search over these alternatives. The present sweep is therefore a post hoc structural ablation rather than a model-selection procedure.

Table H.1 shows the expected trade-off. Increasing K gives the conditioner greater freedom to adapt the physical RBF geometry and generally reduces the mean prediction error, but it also increases the number of task-dependent geometric outputs and hence the trainable parameter count. At one extreme, $K = 50$ and $M_{\text{child}} = 4$ use 127 054 trainable parameters and achieve a mean relative L^2 error of 0.228%. At the other, $K = 10$ and $M_{\text{child}} = 20$ reduce the trainable parameter count to 96 854, a reduction of approximately 23.8%, while retaining a mean error of 0.302%. The absolute increase in mean error across these two configurations is only 0.074 percentage points.

The configuration used in the main parametric Burgers experiment, $K = 25$ and $M_{\text{child}} = 8$, contains 108 179 trainable parameters, approximately 14.9% fewer than the $K = 50$ configuration, while achieving a mean relative L^2 error of 0.270%. These results support the intended role of the parent–child construction: it provides a controllable reduction in independently generated geometry, and consequently in conditioner size, while preserving the full set of $N = 200$ physical RBFs and their independent expansion coefficients.

We therefore do not interpret the lowest-error configuration in Table H.1 as an optimized operating point. Rather, the ablation demonstrates that K controls a practical trade-off between parameter efficiency and geometric flexibility. Smaller K imposes stronger sharing among physical kernels and reduces the number of trainable parameters, whereas larger K relaxes this sharing and provides additional task-adaptive geometric freedom.

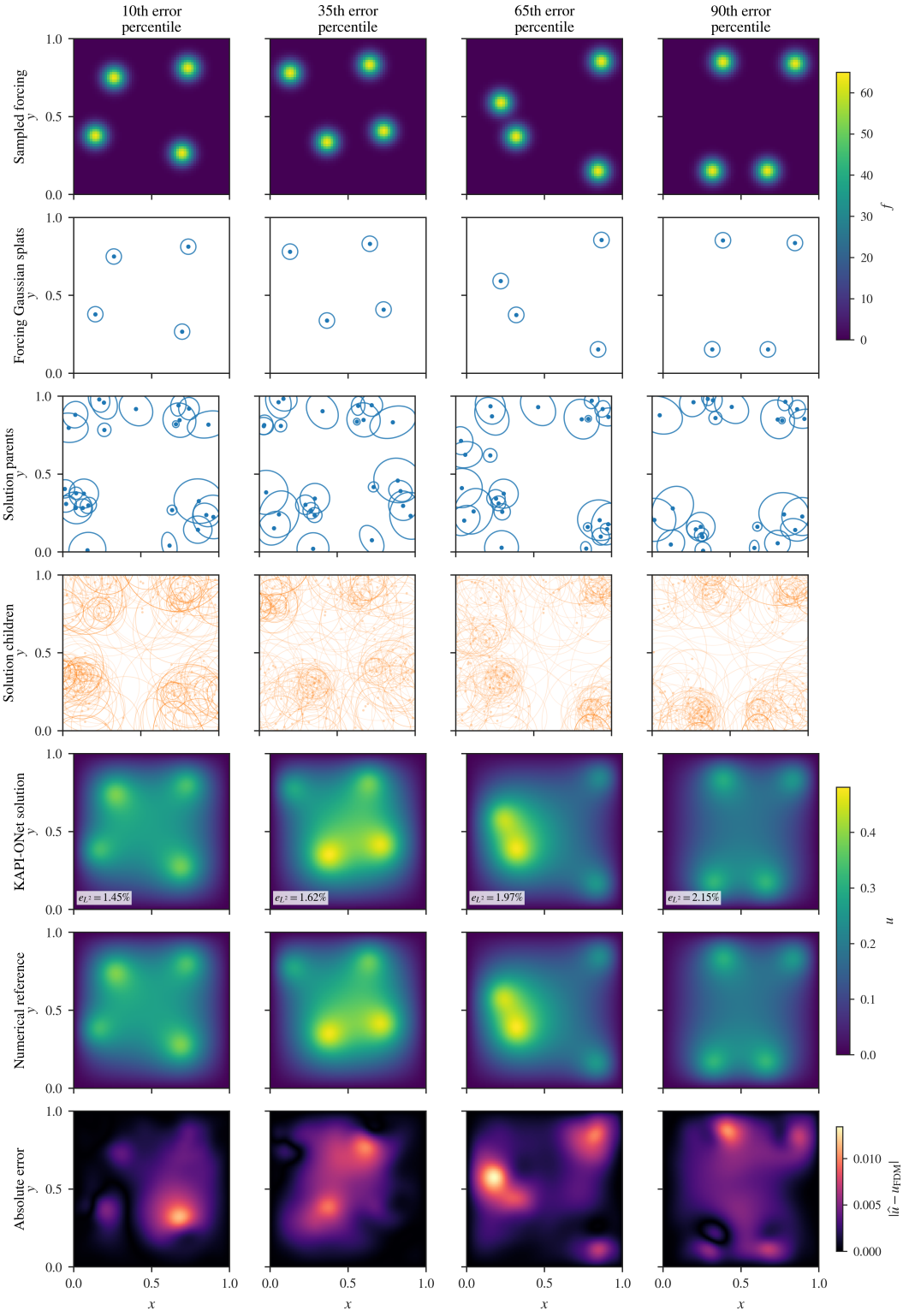

Figure E.2: KAPI-ONet representations and predictions for four held-out Poisson problems with localized Gaussian forcing, selected at the 10th, 35th, 65th, and 90th percentiles of the test-error distribution. The compact forcing representation is mapped to task-conditioned parent and child Gaussian RBFs representing the solution field.

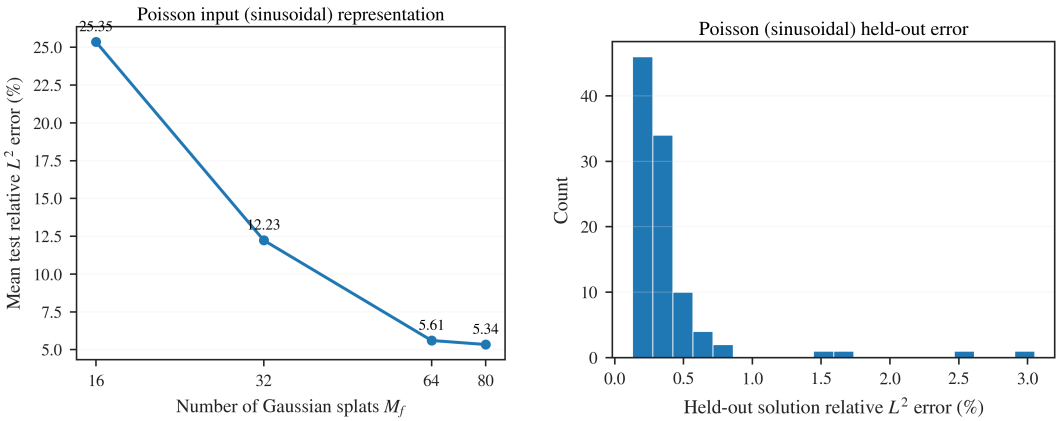

Figure E.3: Sinusoidal-forcing Poisson benchmark results. Left: GSFAE capacity study for the forcing representation; the downstream operator uses $M_f = 64$ Gaussian splats. Right: distribution of the relative L^2 solution error over the 100 held-out Poisson problems.

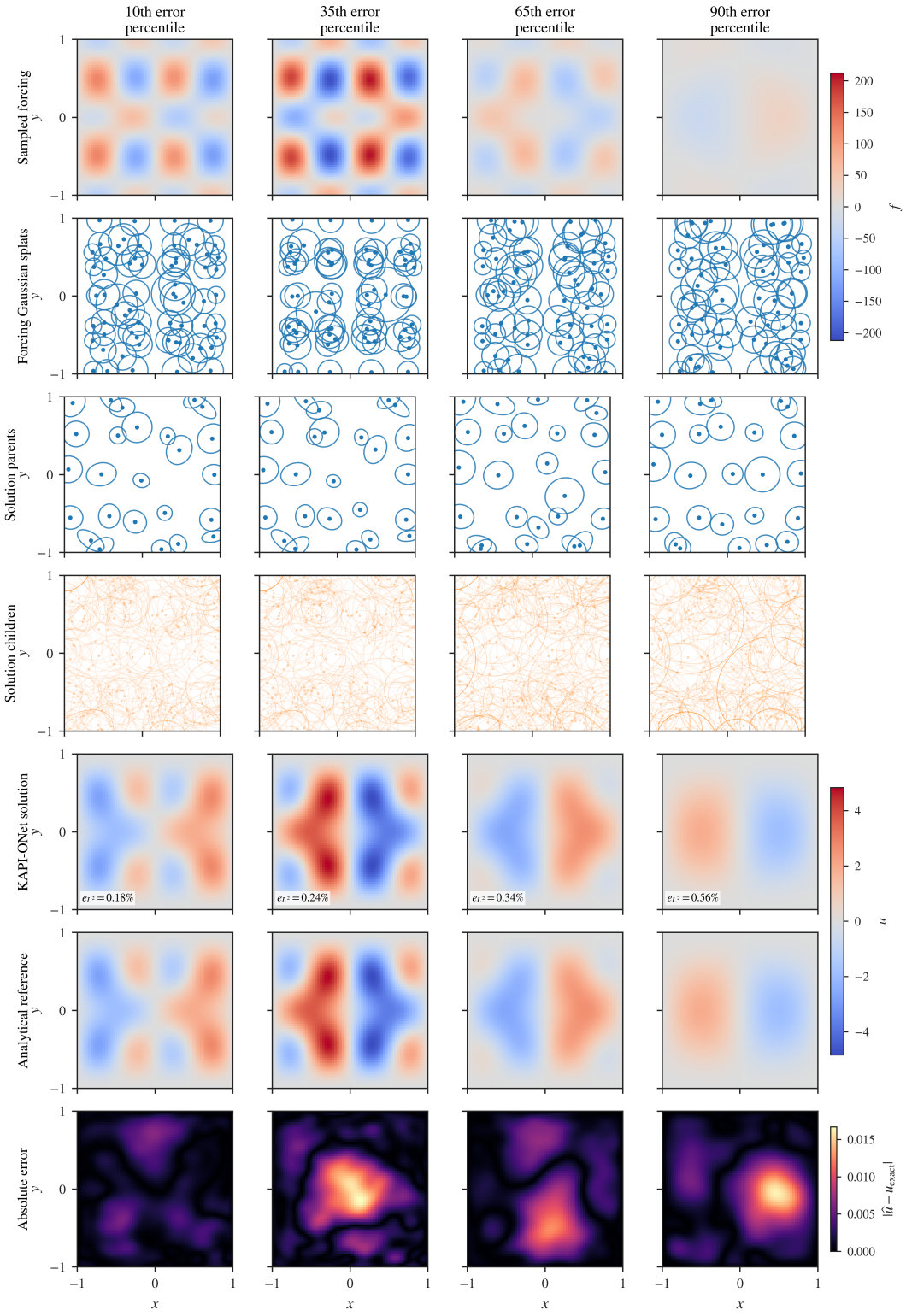

Figure E.4: KAPI-ONet representations and predictions for four held-out Poisson problems with sinusoidal forcing, selected at the 10th, 35th, 65th, and 90th percentiles of the test-error distribution.

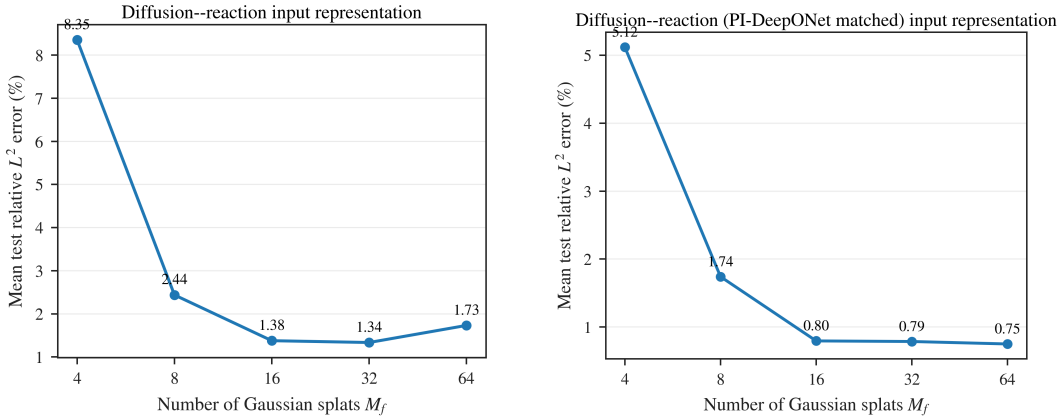

Figure E.5: GSFAE capacity studies for the diffusion–reaction forcing functions. Left: original KAPI-ONet configuration with 2000 operator-training functions. Right: PI-DeepONet-matched configuration with 10 000 operator-training functions.

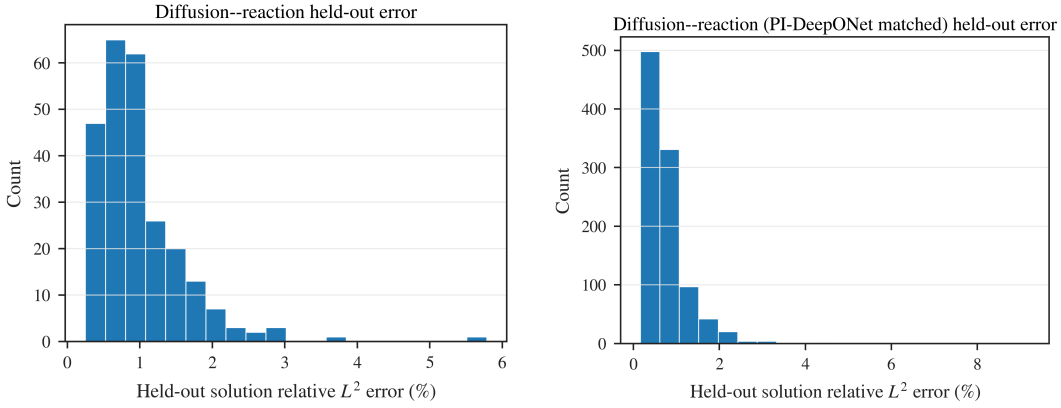

Figure E.6: Held-out relative L^2 solution-error distributions for the diffusion–reaction benchmark. Left: original KAPI-ONet configuration evaluated on 250 held-out forcing functions. Right: PI-DeepONet-matched configuration evaluated on 1000 held-out forcing functions.

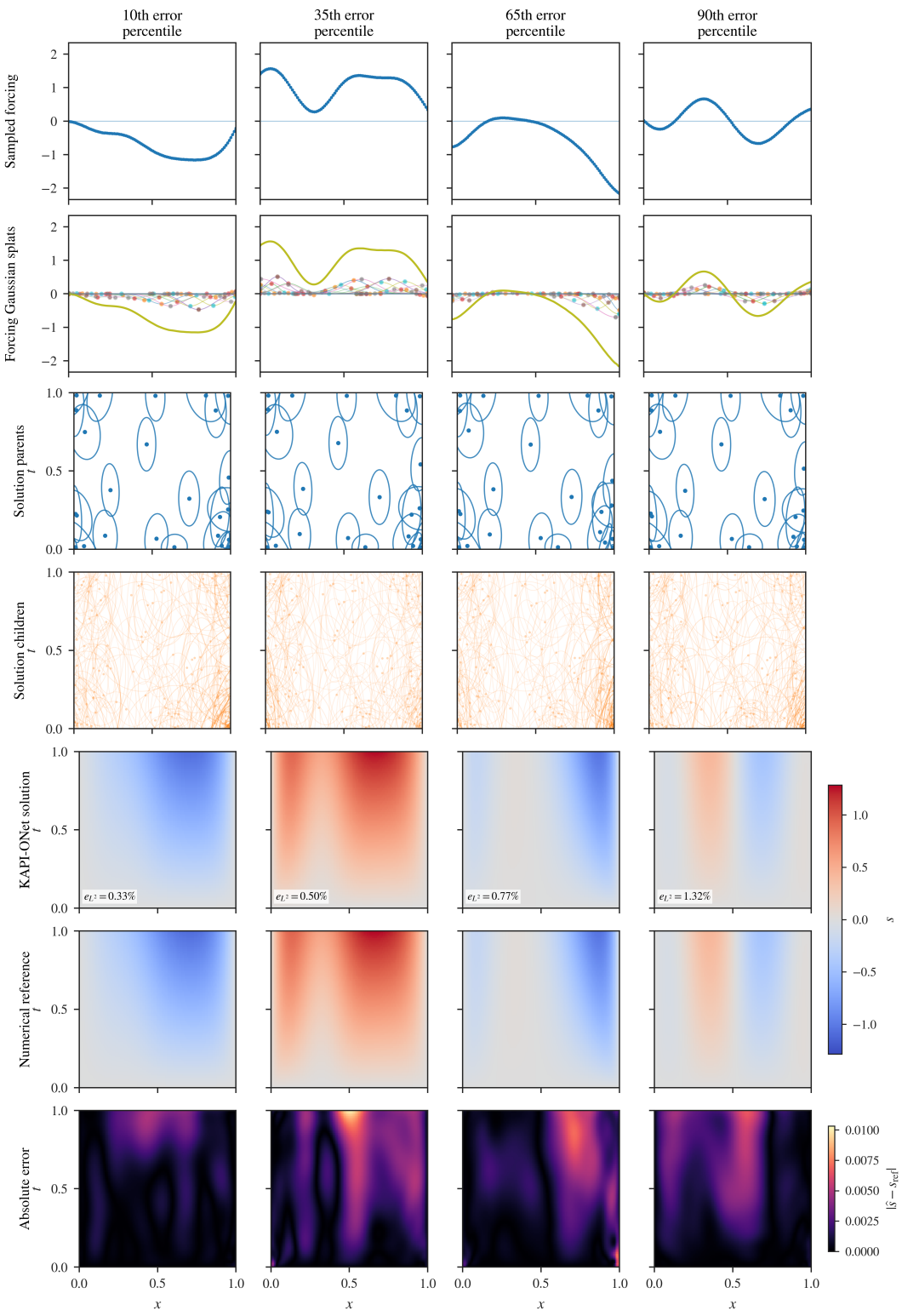

Figure E.7: KAPI-ONet representations and predictions for four held-out diffusion–reaction problems under the PI-DeepONet-matched configuration, selected at the 10th, 35th, 65th, and 90th percentiles of the test-error distribution.

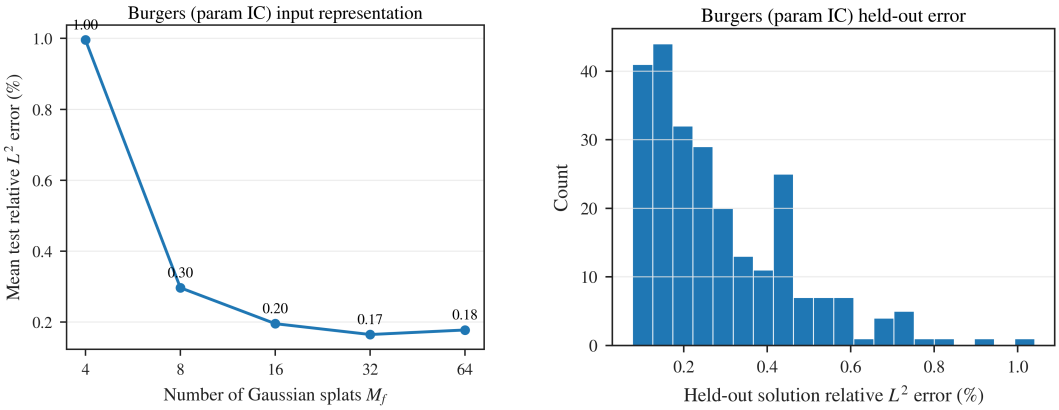

Figure E.8: Parametric Burgers benchmark results. Left: GSFAE capacity study for the initial-condition representation; the selected model uses $M_f = 32$ Gaussian splats. Right: distribution of the relative L^2 solution error over the 250 held-out problems.

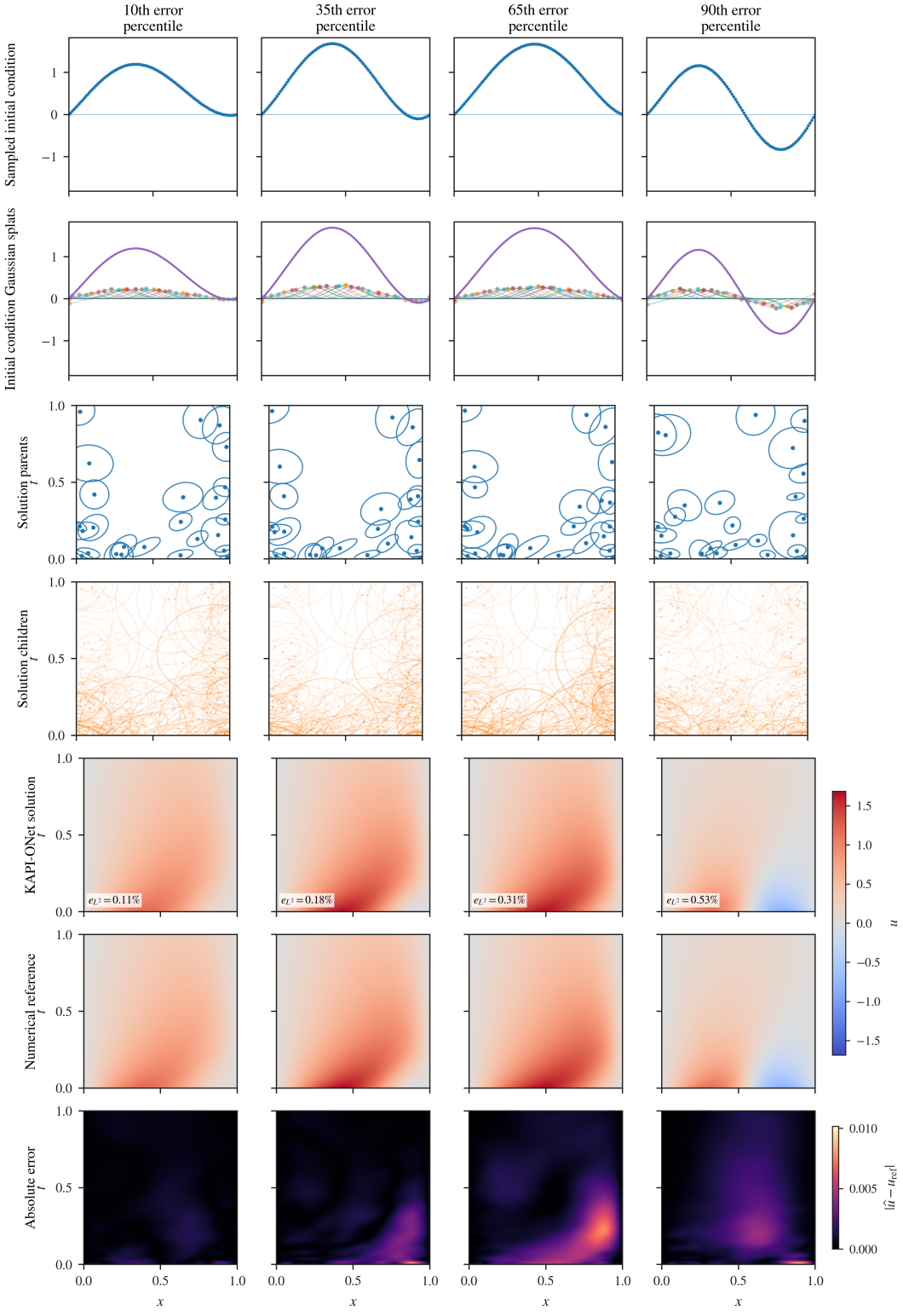

Figure E.9: KAPI-ONet representations and predictions for four held-out parametric Burgers problems selected at the 10th, 35th, 65th, and 90th percentiles of the test-error distribution.

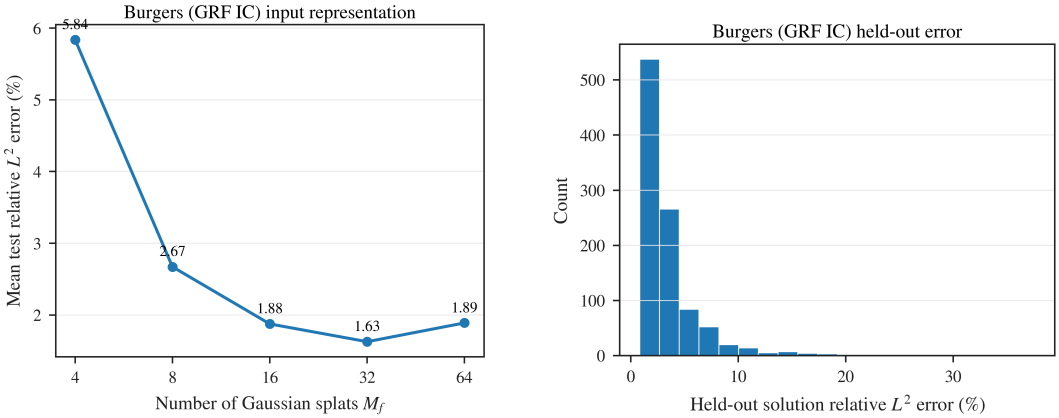

Figure E.10: GRF Burgers benchmark results. Left: GSFAE capacity study for the initial-condition representation; the selected model uses $M_f = 32$ Gaussian splats. Right: distribution of the relative L^2 solution error over the 1000 independently held-out GRF Burgers problems.

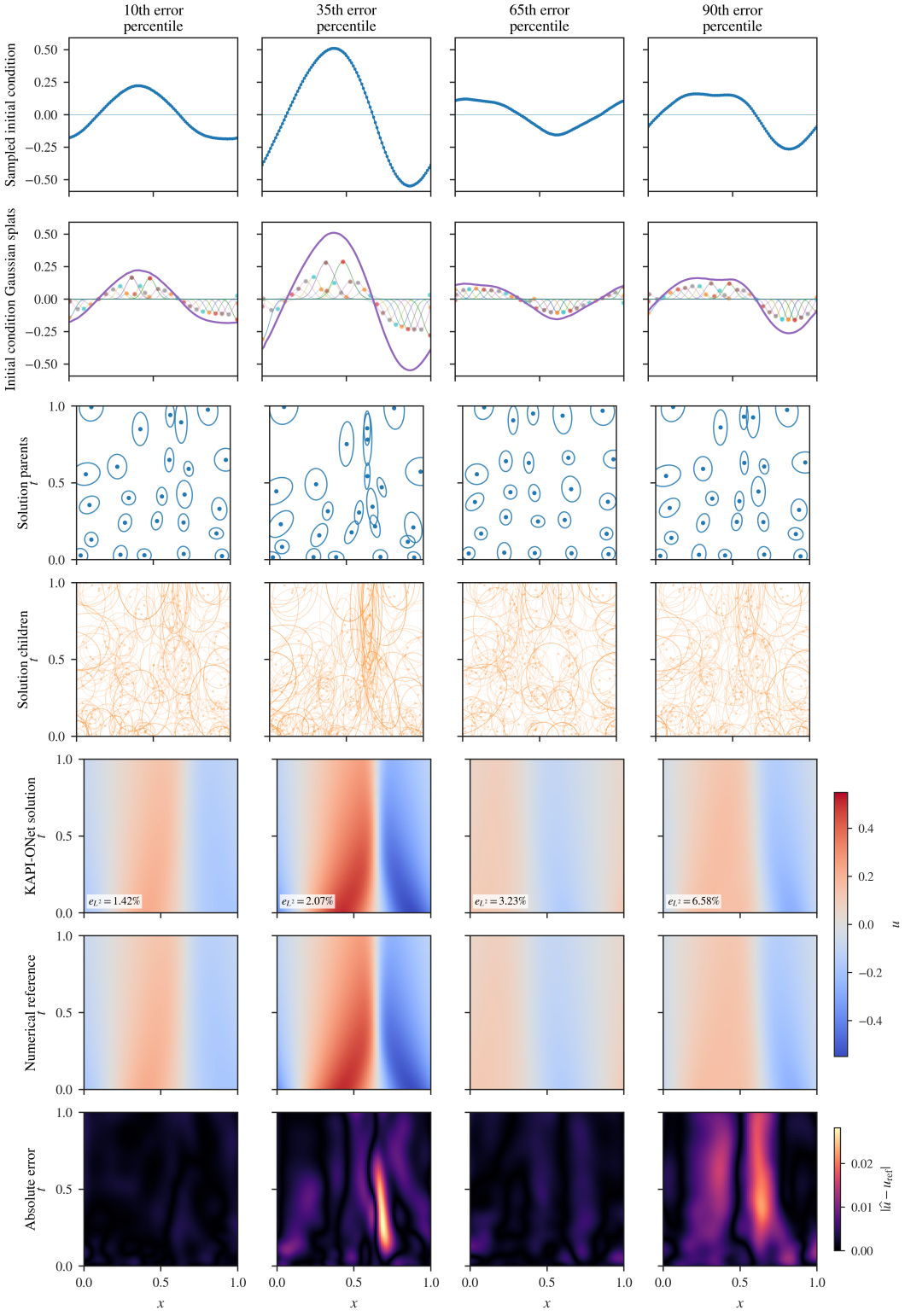

Figure E.11: KAPI-ONet representations and predictions for four held-out GRF Burgers problems selected at the 10th, 35th, 65th, and 90th percentiles of the test-error distribution.

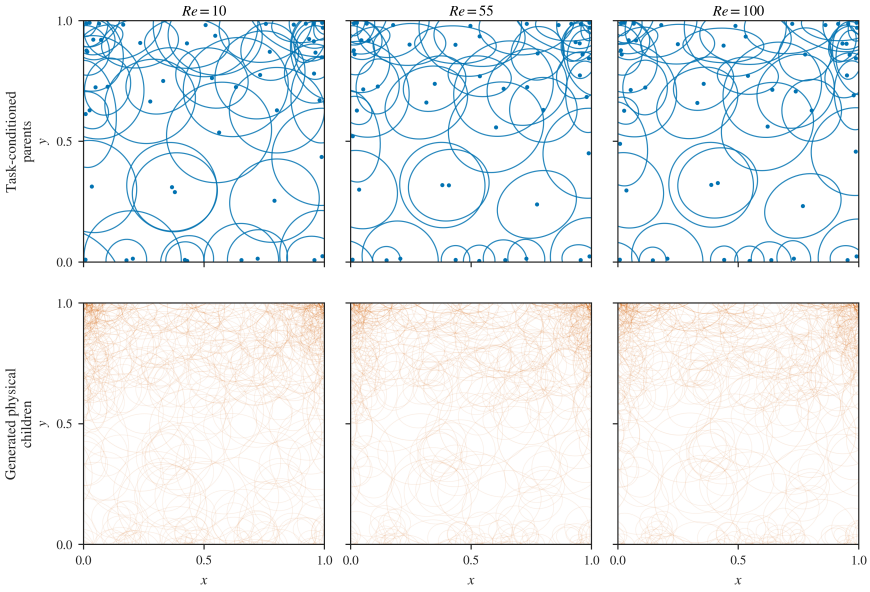

Figure E.12: Task-conditioned KAPI-ONet geometry for the lid-driven cavity at $Re = 10$, 55, and 100. Top: reference parents after Reynolds-dependent shifts in center, anisotropic scale, and orientation. Bottom: the corresponding $N = 400$ generated physical child RBFs.

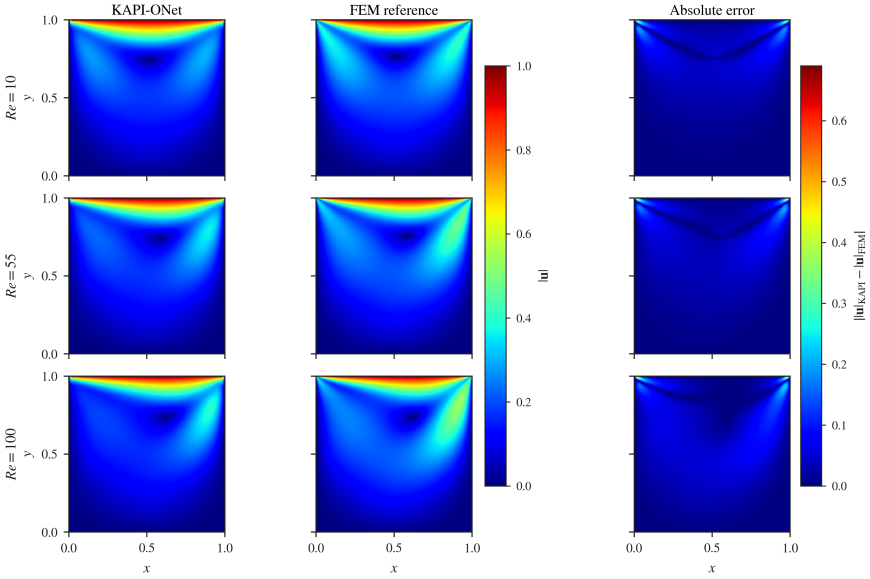

Figure E.13: Velocity-magnitude comparison for the lid-driven cavity at $Re = 10$, 55, and 100. Left: KAPI-ONet prediction. Middle: independent DOLFINx/FEM reference. Right: pointwise absolute error. FEM fields are used only for post-training evaluation.

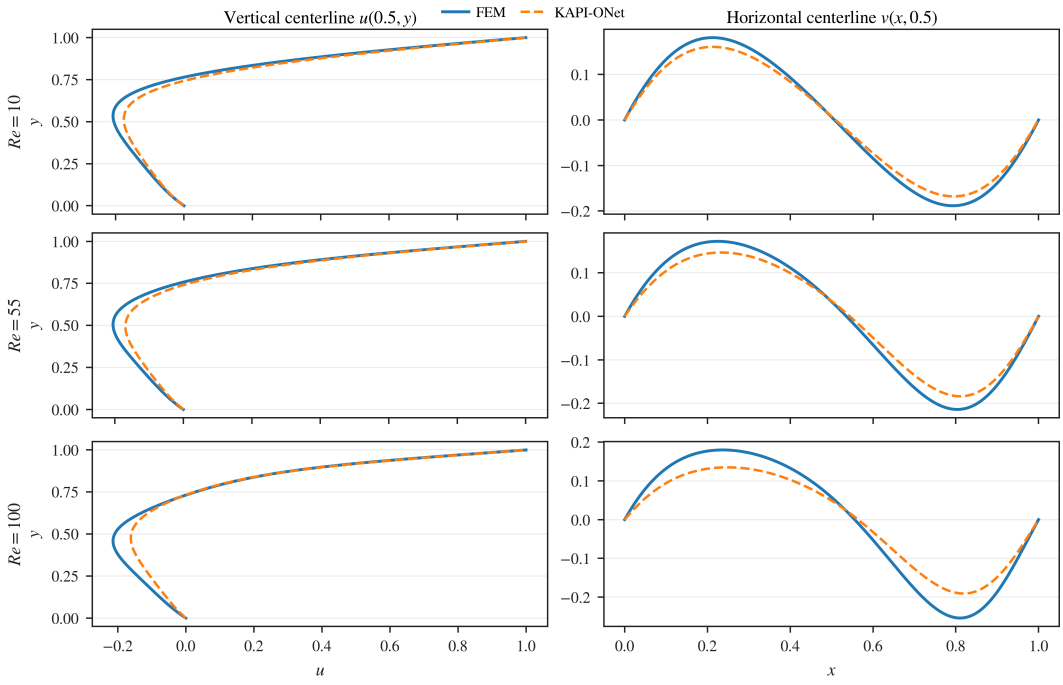

Figure E.14: Centerline velocity profiles for the parametric lid-driven cavity problem. Left: horizontal velocity $u(0.5, y)$ along the vertical centerline. Right: vertical velocity $v(x, 0.5)$ along the horizontal centerline. KAPI-ONet predictions are compared with independent FEM references for $Re = 10$, 55, and 100.

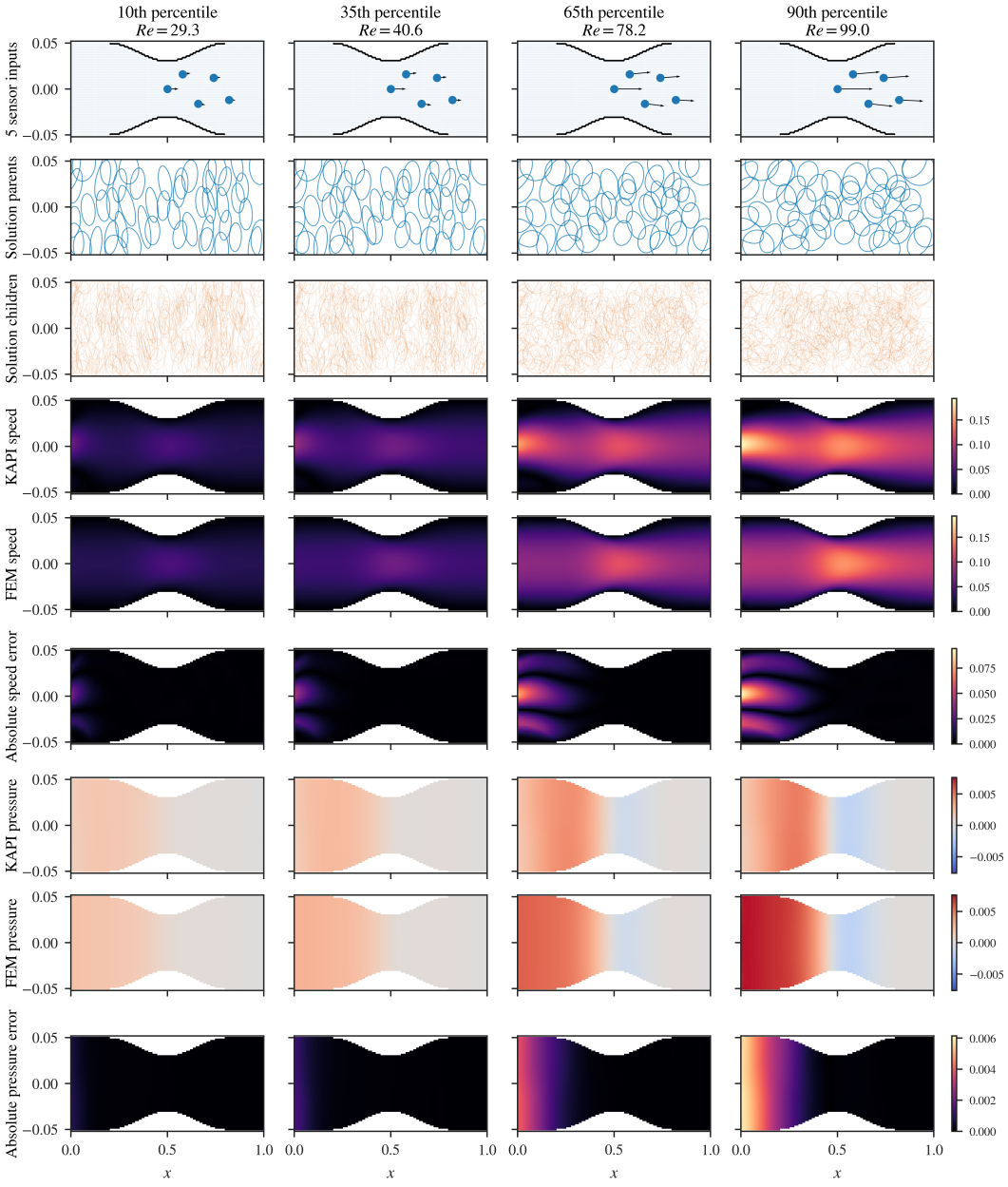

Figure E.15: Sparse-observation inverse-flow reconstruction with KAPI-ONet. Columns show held-out cases selected at the 10th, 35th, 65th, and 90th percentiles of the velocity-error distribution. From top to bottom: the five measured interior velocity vectors, task-conditioned solution parents, generated child RBFs, KAPI-ONet velocity magnitude, FEM velocity magnitude, absolute velocity error, KAPI-ONet pressure, FEM pressure, and absolute pressure error. The task-specific inlet profile and Reynolds number are hidden from the conditioner.

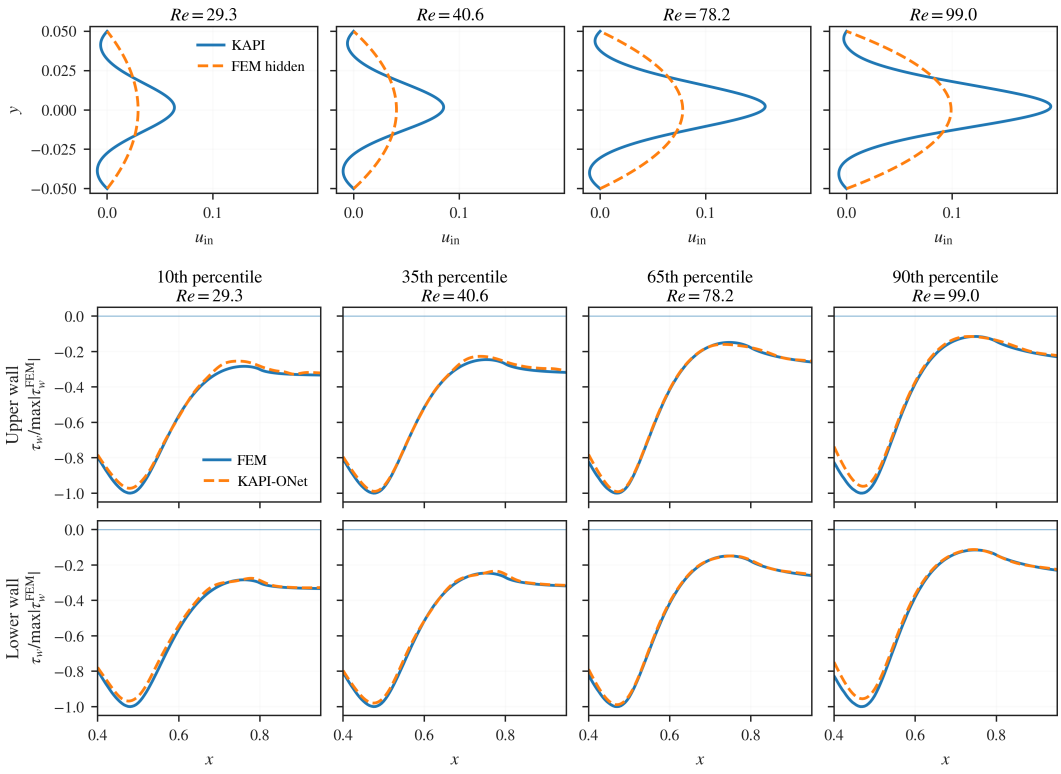

Figure E.16: Non-identifiability of the hidden inlet condition versus recovery of the downstream quantity of interest. Top: inlet velocity profiles inferred retrospectively from the KAPI-ONet flow representation compared with the hidden FEM inlet profiles for four representative held-out cases. The true inlet condition is not supplied during training or inference. Bottom: normalized signed wall shear stress on the upper and lower walls over $x \in [0.40, 0.95]$ for the same cases. Despite the inaccurate recovery of the hidden inlet profile, KAPI-ONet closely reproduces the FEM wall shear stress in the stenotic and downstream region.

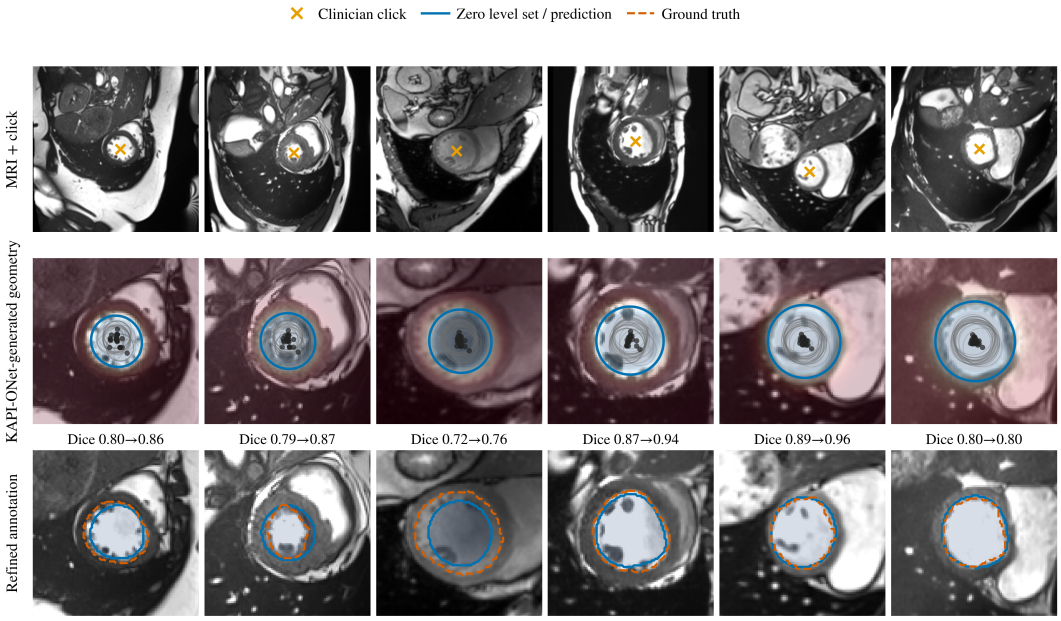

Figure E.17: Task-conditioned weakly supervised cardiac annotation with KAPI-ONet. Top: cardiac MR image and the single weak localization point. Middle: generated 25-RBF geometry and the zero level set of the corresponding implicit field. Bottom: annotation after the fixed mask-free variational correction, compared retrospectively with the expert contour. Representative cases are selected from quantiles of the KAPI-ONet-generated region area rather than from ground-truth accuracy.

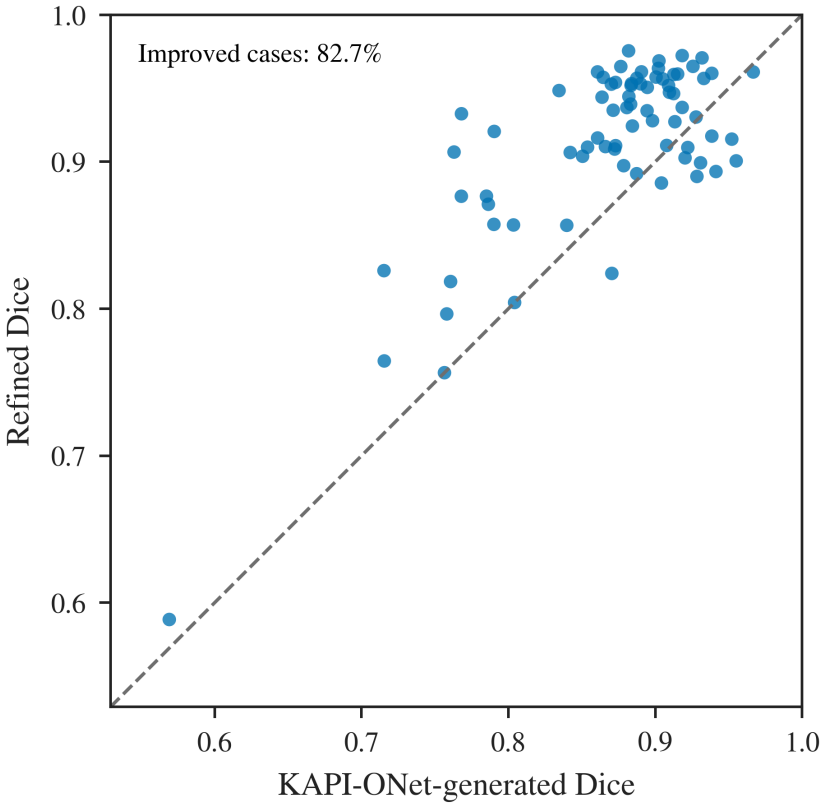

Figure E.18: Effect of the mask-free variational correction on the 75 held-out cardiac MR images. Each point compares the Dice score of the frozen KAPI-ONet prediction with the corresponding corrected annotation; the dashed diagonal denotes no change. The correction improves 82.7% of the test cases. Expert masks enter only the retrospective computation of the plotted Dice values.

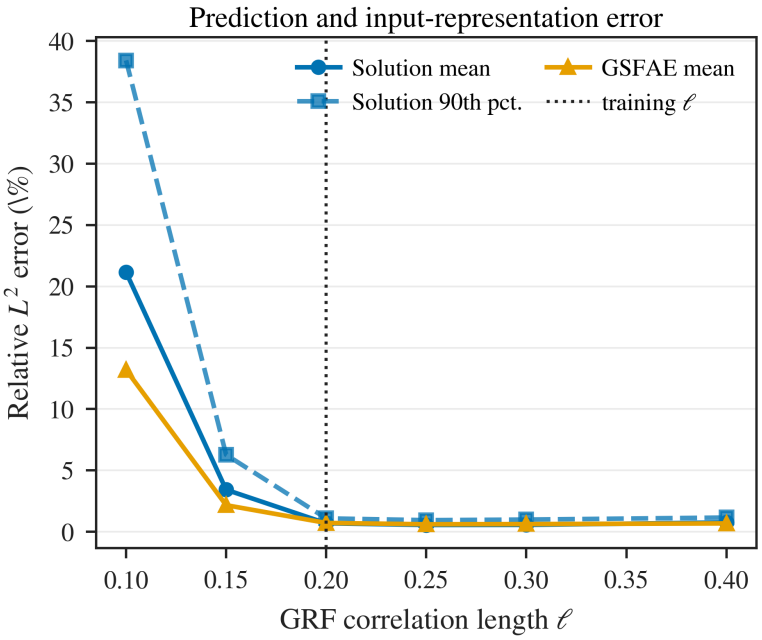

Figure G.1: Frozen-operator generalization under a shift in the GRF correlation length for the diffusion–reaction problem. The dotted line marks the training value $\ell = 0.20$. Curves show the mean and 90th-percentile solution relative L_2 errors together with the mean frozen-GSFAE reconstruction error over 100 tasks at each correlation length.

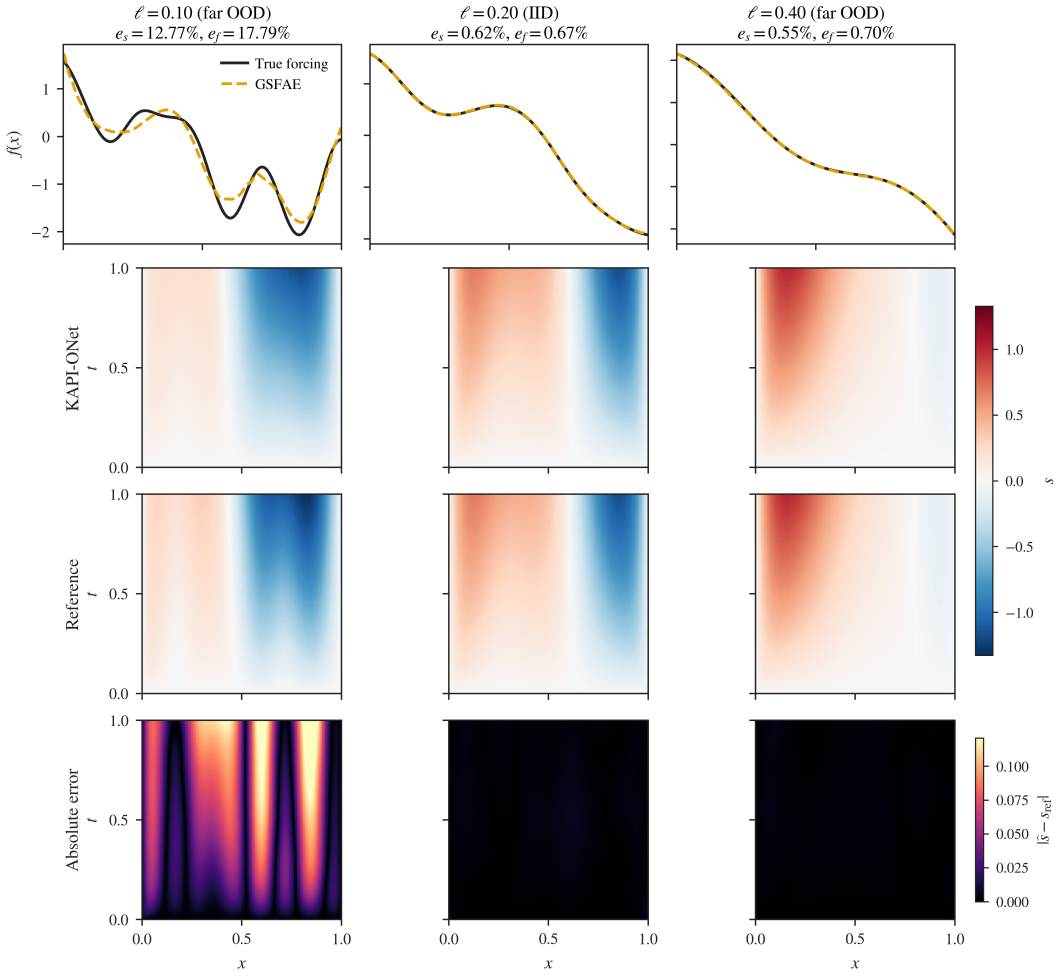

Figure G.2: Representative diffusion–reaction cases under GRF correlation-length shift. The first row compares the true forcing with its frozen-GSFAE reconstruction; the second and third rows show the KAPI-ONet prediction and finite-difference reference solution, respectively; and the final row shows the absolute solution error. Columns correspond to the far-OOD case $\ell = 0.10$, the in-distribution case $\ell = 0.20$, and the far-OOD case $\ell = 0.40$. The same paired Gaussian draw is used in all three columns.

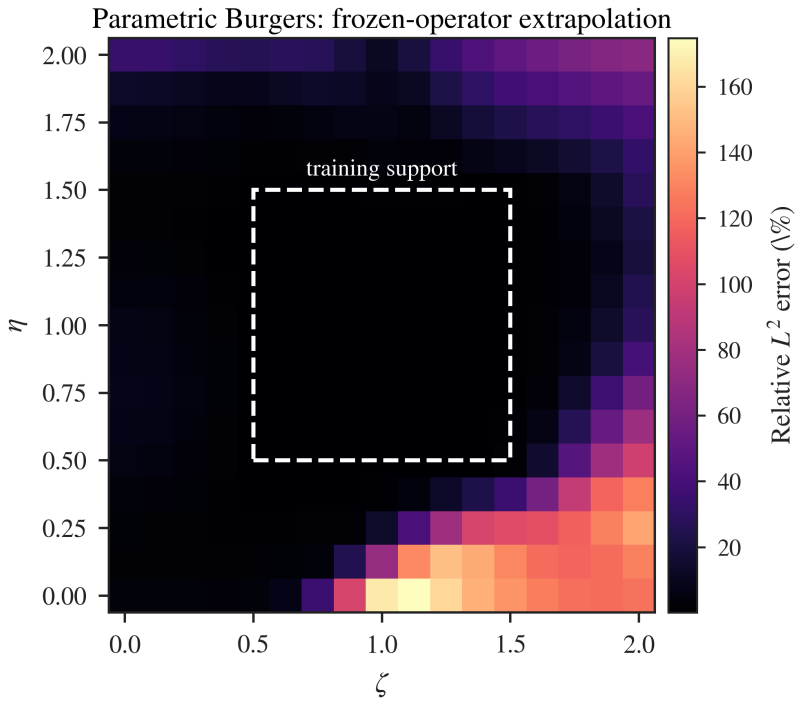

Figure G.3: Frozen-operator parameter extrapolation for the parametric Burgers problem. The dashed rectangle denotes the training support $(\zeta, \eta) \in [0.5, 1.5]^2$. Colors show the relative L_2 error of the predicted space–time solution over a deterministic 17×17 parameter grid spanning $[0, 2]^2$.

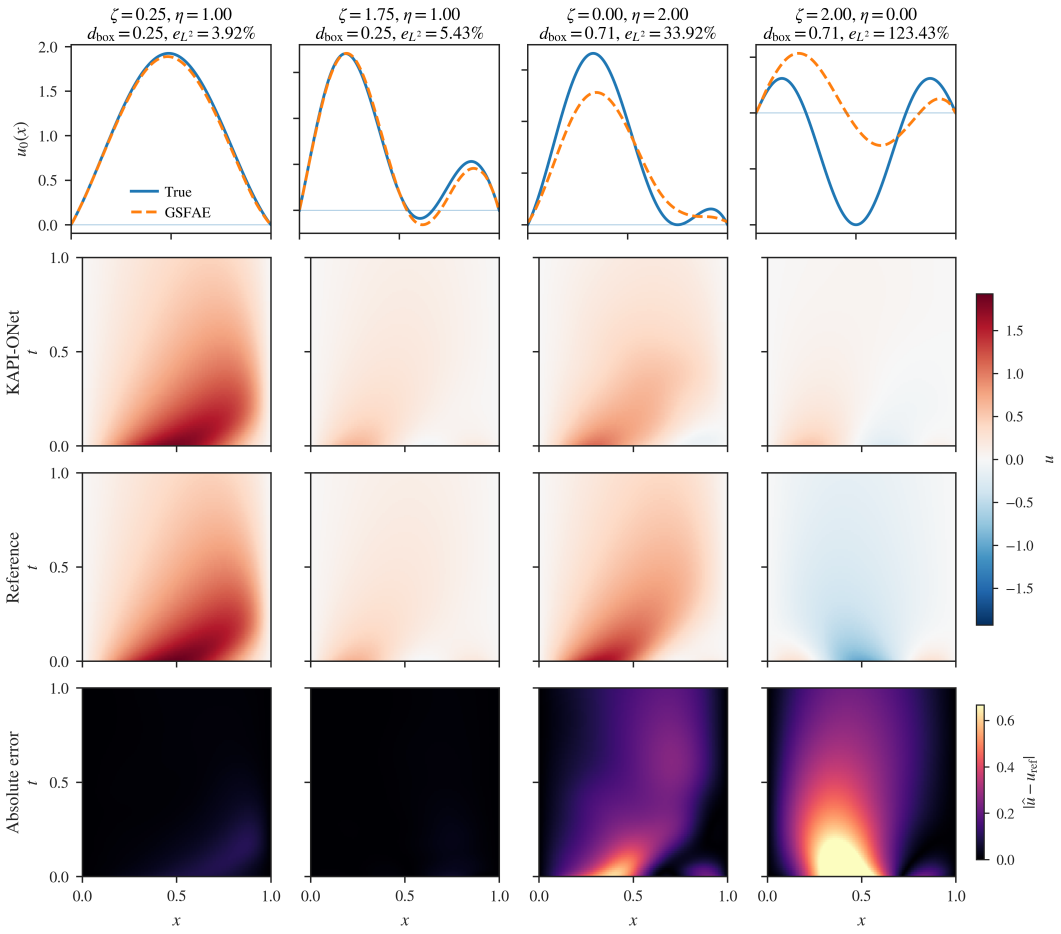

Figure G.4: Representative Burgers extrapolation cases selected a priori from parameter location. The first row compares the true initial condition with its frozen GSFAE reconstruction; the second and third rows show the KAPI-ONet and finite-difference reference solutions, respectively; and the final row shows the absolute solution error. The first two columns are near-OOD cases and the final two are far-OOD corners.

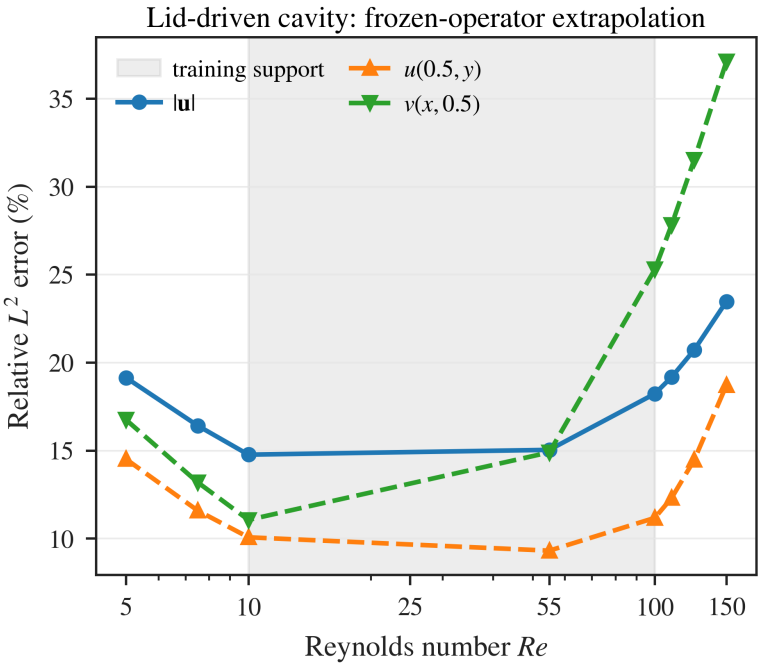

Figure G.5: Frozen-operator Reynolds-number extrapolation for the lid-driven cavity. The shaded region denotes the training interval $Re \in [10, 100]$. Relative L_2 errors are shown for velocity magnitude and the two classical centerline velocity profiles.

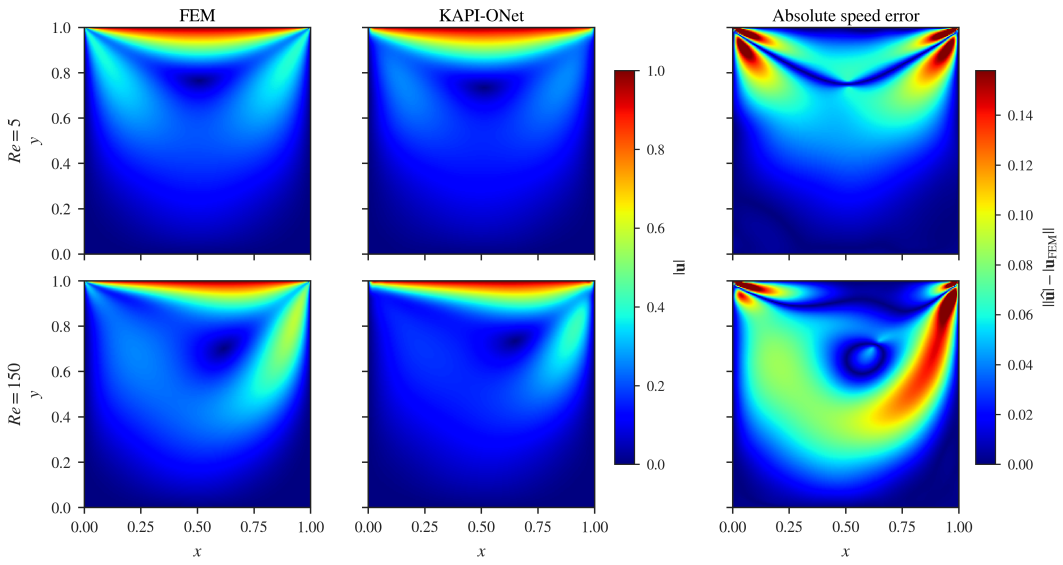

Figure G.6: Velocity-magnitude extrapolation at the two farthest OOD Reynolds numbers. FEM reference fields, frozen KAPI-ONet predictions, and absolute speed errors are shown for $Re = 5$ and $Re = 150$. FEM and KAPI-ONet use a common velocity-magnitude scale within each comparison, while the error column uses a separate scale.