
Distillation of Foundation Models for Time-dependent PDEs

Daniel Musekamp^{1*} Boshra Ariguib^{1,2} Andrei Manolache^{1,3} Mathias Niepert¹

¹University of Stuttgart ²University of Cologne ³Bitdefender

Abstract

Foundation models for time-dependent partial differential equations (PDEs) are trained on large and diverse collections of physical systems and can generalize effectively to new downstream tasks. After fine-tuning on only a few trajectories from a target domain, they can achieve strong accuracy in low-data regimes. However, these models are typically large and computationally intensive, limiting their usefulness as fast surrogates for numerical solvers. We propose Teacher Rollout Extension (TRES), a knowledge distillation framework that transfers the predictive capability of a pretrained foundation model into a compact and efficient student. Starting from a fine-tuned teacher, TRES augments limited downstream data by generating long synthetic trajectories through teacher rollouts, optionally with periodic noise injection. This procedure samples from the teacher-induced rollout distribution without requiring explicit knowledge of the initial-condition distribution, while exposing the student to long-horizon states and local recovery behavior around states encountered during autoregressive prediction. The student can further incorporate task-specific inductive biases, such as equivariance, that the teacher does not necessarily enforce. The experiments show that the student trained with TRES can match or surpass the teacher’s accuracy while achieving more than an order-of-magnitude speedup in inference.

1 Introduction

Processes such as fluid dynamics or cell growth can be described using partial differential equations (PDEs). Since solving PDEs can be slow, recent research has investigated applying neural networks as surrogate models to learn solutions to PDEs. One core problem with neural PDE surrogates is that they require a training dataset, which is often generated using an expensive numerical solver, thereby making the surrogate modeling potentially uneconomical. As a remedy, foundation models (FMs) for PDEs have been introduced [9, 28, 29]. By pretraining a surrogate on a large set of diverse physics, FMs can transfer knowledge to downstream tasks with fewer training samples. On the downside, inference for such models is computationally expensive, as they require more parameters to generalize well. Slow inference prohibits their use in applications that have real-time requirements, such as planning and control tasks in manufacturing or energy systems [12, 41]. Furthermore, the used architectures are mostly large transformers that lack inductive biases, since these may vary from problem to problem. Therefore, we propose using knowledge distillation (KD, [4, 10]) to reconcile the trade-off between fast inference and strong generalization (Fig. 1). Specifically, the pretrained FM is first fine-tuned on a small set of trajectories in the target domain. Afterward, the FM is used as the teacher to amend the available data with additional information. Research on KD for PDEs has been limited and has not considered the FM regime. Additionally, most prior work relies on specialized architectures or assumes full knowledge of the initial state distribution [17, 45]. However, access to

*Corresponding author: daniel.musekamp@ki.uni-stuttgart.de

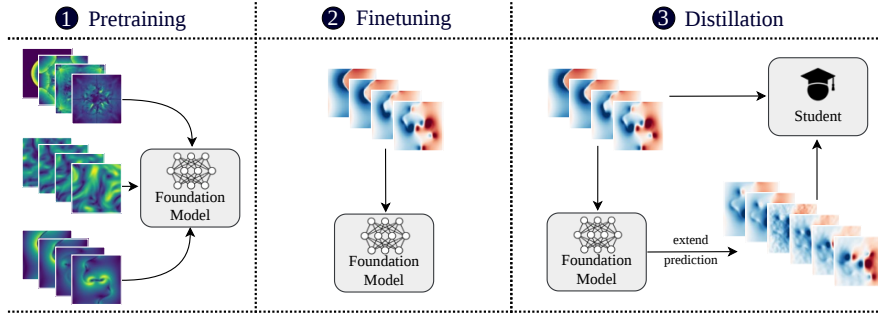


Figure 1: Foundation models are pretrained on a variety of PDE prediction tasks first (1), enabling them to adapt to a new task with only a few trajectories (2). To overcome their high inference time, we propose to distill the foundation model into a smaller and potentially equivariant student by extending the training trajectories using noisy rollouts of the teacher (3).

the downstream initial-condition distribution is not always available, e.g., when data originates from experiments or partially observed systems. Furthermore, these methods are not applicable to FMs that require multiple context frames, since generating valid contexts requires additional simulator calls. See App. D for a full discussion of related work.

Hence, we introduce Teacher Rollout Extension (**TREX**), a distillation technique that avoids sampling new initial conditions. Starting from the few available downstream trajectories, TREX generates long teacher rollouts and trains the student on the resulting teacher-labeled trajectories. This exposes the student to the teacher-induced rollout distribution, i.e., the distribution of states encountered during autoregressive prediction, rather than only the short trajectories observed in the downstream data. Periodic noise injection further broadens local state-space coverage and teaches the student how the teacher evolves from perturbed states. To transfer the teacher’s knowledge to the student model, we add these teacher-generated trajectories to the student dataset and train on a mixture of ground-truth and teacher-labeled transitions. To overcome the aforementioned lack of inductive biases in FMs, we propose to distill the teacher into a student with the appropriate inductive biases, such as equivariances of the downstream task. Including equivariance in the model architecture can increase data efficiency [7, 46]. We investigate the effectiveness of our method through experiments with the Poseidon [9] and Walrus [29] foundation models on a number of fluid dynamics tasks, showing that TREX can achieve teacher accuracy while drastically reducing the number of parameters and inference time.

2 Knowledge Distillation of PDE Foundation Models

We consider autoregressive neural surrogates for time-dependent PDEs. Given a small downstream dataset $\mathcal{D}_{\text{GT}} = \{(\mathbf{u}_n^0, \dots, \mathbf{u}_{N_T-1}^n)\}_{n=1}^N$, our goal is to distill a fine-tuned foundation model T into a compact student S . The teacher predicts $\mathbf{z}_{t+1} = T(\mathbf{z}_t)$; for multi-frame models, $\mathbf{z}_t$ denotes the complete context window. Each trajectory consists of N_T states. Our goal is to build an accurate and efficient surrogate model by infusing knowledge from a foundation model (FM) used as the teacher $\mathcal{T}$, that autoregressively predicts the next state solution $\mathbf{z}^{t+1} = \mathcal{T}(\mathbf{z}^t)$. In the first step, the FM is fine-tuned on the available downstream task data. To distill the fine-tuned model into the student S , TREX not only trains the student on the available ground-truth downstream trajectories, but also generates additional training data by rolling out the teacher from available states. Hence, TREX constructs an extended teacher-induced state occupancy measure:

$$\rho_{\mathcal{T}}^K = \frac{1}{K} \sum_{t=0}^{K-1} (\mathcal{T}^t)_{\#} \nu_0, \quad \nu_0 = \frac{1}{N} \sum_{n=1}^N \delta_{\mathbf{u}_n^0}. \quad (1)$$

Here, $\delta_{\mathbf{u}_n^0}$ is the Dirac measure concentrated at $\mathbf{u}_n^0$, ν_0 the observed starting ground truth states, $(\mathcal{T}^t)_{\#} \nu_0$ denotes the pushforward of ν_0 under t autoregressive teacher steps and K is the number of rollout steps. Hence, the occupancy measure describes the state distribution induced by rolling out the teacher from the selected set of ground truth samples. The corresponding distillation objective trains

the student on transitions sampled from the measure. Thus, TREX does not require access to the true initial condition distribution. Instead, it reuses the few ground-truth trajectories as anchors and expands them through the dynamics learned by the FM. This perspective connects TREX to the long-time behavior of dynamical systems. Many dissipative PDEs and chaotic systems possess recurrent long-time structure, such as attractors, so that trajectories initialized from different states eventually visit similar regions of the state space. If the teacher dynamics admit an invariant measure $\mu_{\mathcal{T}}$ satisfying $\mathcal{T}_{\#} \mu_{\mathcal{T}} = \mu_{\mathcal{T}}$, then the finite occupancy measure $\rho_{\mathcal{T}}^K$ can be viewed as an approximation to the teacher’s long-time rollout distribution. This is especially relevant to ergodic systems, where different starting points eventually cover the same state distribution (see Fig. 3 in App. G).

Noisy Teacher Rollouts. Deterministic rollouts sample states on the teacher trajectory, but may provide limited coverage of nearby off-trajectory states. Additionally, it can be beneficial to include data points near typical rollout states in order to teach the student how the dynamics evolve from perturbed inputs. Hence, we inject noise periodically during the teacher rollout

$$\tilde{z}_t = z_t + m_t \xi_t, \quad z_{t+1} = T(\tilde{z}_t), \quad m_t = \mathbf{1}\{t > 0 \wedge t \equiv 0 \pmod{T_{\text{Noise}}}\}, \quad (\xi_t)_c(x) \sim \mathcal{N}(0, \sigma^2 s_c^2).$$

Here, ξ^t is Gaussian noise applied independently at each spatial location and channel, and s_c denotes the global standard deviation of channel c computed over the dataset, and σ is a fixed scaling factor. The noise is not applied at every step, but at intervals of length T_{Noise} , so that we obtain unperturbed sub-trajectories for training and allow the teacher to converge back to the attractor if the noise leads to a state far off. Adding noise during training is a common strategy for autoregressive neural PDE solvers [33, 36, 40]. However, these approaches perturb only the input states, leaving the target trajectory unchanged. As a result, the surrogate learns to map perturbed states back to known states and is exposed to slightly out-of-distribution states, but the correction is systematically biased, since the true dynamical system would generally evolve differently from the perturbed state. By performing noisy rollouts from all available ground truth trajectories, we obtain the dataset $\mathcal{D}_{\text{TREX}}$, which consists of short sub-trajectories defined by the segments of the trajectory between noise applications. The final student objective combines ground-truth supervision with the teacher rollout distillation, $\mathcal{L}_S = \mathcal{L}_{\text{GT}} + \lambda_{\text{TREX}} \mathcal{L}_{\text{TREX}}$. The losses on both datasets are computed on k -step sub-trajectories.

Enforcing Physical Constraints. PDE FMs are usually constructed using large transformer architectures (App. D). This makes them flexible enough to model many different systems, but it also means that they often do not explicitly encode the symmetries of a specific downstream PDE. Once the downstream task is fixed, however, such a structure may be known and can be built into the student. We focus on equivariance as one such constraint. Let G be a group of transformations acting on the state space, and let $\tau_g u$ denote the transformed state for $g \in G$. A model $\mathcal{M}$ is equivariant to G if $\forall g \in G: \mathcal{M}(\tau_g u) = \tau_g \mathcal{M}(u)$. For a surrogate, this means that transforming the input state and then predicting the next state gives the same result as predicting first and then transforming the output. Many PDEs satisfy such symmetries. For example, on a periodic domain, the solution operator of a translation-invariant PDE satisfies $\Phi_{\Delta t}(\tau_a u) = \tau_a \Phi_{\Delta t}(u)$, where τ_a denotes a spatial shift. This provides a complementary benefit to distillation. The teacher contributes knowledge acquired during large-scale pretraining, while the student can impose task-specific structure known for the PDE.

3 Experiments

For our main experiments, we use the Poseidon-L model [9] as the teacher. The teacher is distilled into a lightweight TFNO [15, 16], which represents FNO’s [18] weight matrix using a Tucker factorization to reduce the number of parameters but also improve generalization through regularization effects. The student is trained autoregressively with 2-step sub-trajectories. More model and training details are listed in App. F. For TREX, the teacher is rolled out for 100 steps starting from the initial condition, where noise is injected after every 10 steps. The relative noise standard deviation is set to $\sigma = 1$. As baselines, we compare the original Poseidon-L teacher and the student trained only with the available ground truth data. Additionally, we add relational KD [32] as a feature-based KD method. Lastly, we add a baseline with full knowledge of the true initial condition distribution (IC-KD) similar to [45]. As downstream tasks, we use compressible Euler and incompressible Navier-Stokes equations from [9]. App. E lists the used PDEs in detail. We restrict ourselves to spatiotemporal tasks from Poseidon and exclude tasks where we did not find a clear advantage of the FM teacher over the TFNO student trained from scratch, since we focus on investigating scenarios where a pre-trained FM can successfully transfer its knowledge.

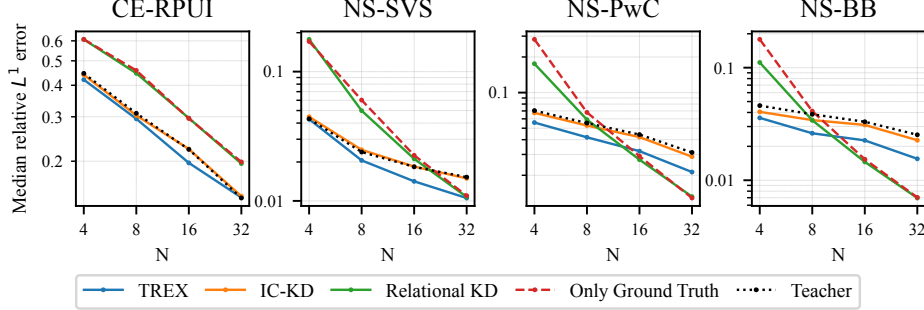


Figure 2: The median relative L^1 error (Eq. (24)) for Poseidon downstream experiments. The distillation methods are compared to the teacher’s accuracy over the number of trajectories in the training set N . TREX manages to match or even surpass the teacher on all tasks.

Fig. 2 shows the main results for the Poseidon distillation. TREX matches or even surpasses its teacher for all considered settings. At the same time, inference speed (Tab. 1) is increased by over an order of magnitude. These findings generalize to different students (App. H.1.1). IC-KD is also able to match the teacher on all investigated settings; however requires the full IC distribution to be known, which is not the case for real-world data and might also be expensive. For example, for NS-BB, the IC is actually an already evolved state (App. E). While relational KD can also slightly improve the student, we were not able to match the teacher when the teacher outperformed ground truth learning.

Table 1: Inference time, GPU memory, and number of parameters for the teacher and student for a batch size of 64.

Model	Time	Memory	# Param.
Poseidon	1.88 s	3749 MB	628.6M
TFNO	0.15 s	1171 MB	174.4K

A reason might be the difficulty of meaningful distance measures in the large-dimensional feature fields encountered in the neural PDE solver latent space. Interestingly, we found that the TFNO is an exceptionally well-performing model for the considered tasks, even capable of outperforming the pretrained Poseidon in certain settings without KD. In these cases, TREX can perform worse than only using the ground truth since the added teacher-generated data is moving the student away from the better solution it would have found using only ground truth. For the other student models (App. H.1.1), we did not find similar results. Thus, KD should only be applied if the FM was found to generalize well to the task at hand using a held-out validation set. Analyzing the error over time shows that the student initially has a larger error but a slower error growth afterward (Fig. 5 in App. H). We also investigate another teacher model (App. H.1.2). For the multi-frame Walrus FM [29] on Kolmogorov Flow [19], TREX reduces the 60-step rollout VRMSE from 0.547 for Walrus to 0.498, while a student trained only on ground truth reaches 0.766 (Tab. 6).

The ablations in App. H further indicate that the gain does not arise solely from mixing in ground-truth data. On NS-PwC with four trajectories, TREX obtains a median relative L^1 error of 0.056, while training only on the teacher-generated TREX trajectories still achieves 0.060, compared with 0.070 for the teacher. Removing noise as well increases the error to 0.064, indicating that both rollout extension and noise injection contribute to the improvement. Regarding equivariance, we found that applying random spatial shifts to the (shift-equivariant) PDE trajectories increased Poseidon’s error from 0.082 to 0.240 for NS-SVS, while the student’s error remained constant due to its equivariant architecture at 0.076 (App. H.3). This shows that enforcing equivariance in the student architecture can improve robustness to out-of-distribution states.

4 Conclusion

We introduce TREX, a simple and effective knowledge distillation framework for PDE foundation models. The results show that compact student models can match the accuracy of large teacher models while achieving substantial improvements in inference speed and memory efficiency. In the future, TREX could be combined with other distillation techniques to also provide more information on the influence of other factors, such as physical parameters, to the student.

Acknowledgments and Disclosure of Funding

Funded by the German Federal Ministry of Research, Technology and Space (BMFTR) as part of the InnoPhase project (funding code: 02NUK078). Additionally, we acknowledge the support of Deutsche Forschungsgemeinschaft (DFG) under funding code 569019417 and the support of the Stuttgart Center for Simulation Science (SimTech). Andrei Manolache acknowledges funding by the EU Horizon project ELIAS (No. 101120237). The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Daniel Musekamp, Andrei Manolache, and Mathias Niepert.

References

- [1] Ishan Amin, Sanjeev Raja, and Aditi S Krishnapriyan. Towards fast, specialized machine learning force fields: Distilling foundation models via energy Hessians. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [2] Lei J Ba and Rich Caruana. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27, 2014.
- [3] X Cheng, Y He, Y Yang, X Xue, S Cheng, D Giles, X Tang, and Y Hu. Learning chaos in a linear way. In *International Conference on Learning Representations*, pp. 55276–55318. ICLR, 2025.
- [4] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International journal of computer vision*, 129(6):1789–1819, 2021.
- [5] Jayesh K. Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized PDE modeling. *Trans. Mach. Learn. Res.*, 2023.
- [6] Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. Dpot: auto-regressive denoising operator transformer for large-scale pde pre-training. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 17616–17635, 2024.
- [7] Jacob Helwig, Xuan Zhang, Cong Fu, Jerry Kurtin, Stephan Wojtowysch, and Shuiwang Ji. Group equivariant fourier neural operators for partial differential equations. In *International Conference on Machine Learning*, pp. 12907–12930. PMLR, 2023.
- [8] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.
- [9] Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes, 2024. URL <https://arxiv.org/abs/2405.19101>.
- [10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [11] David Holzmüller, Viktor Zaverkin, Johannes Kästner, and Ingo Steinwart. A framework and benchmark for deep batch active learning for regression. *Journal of Machine Learning Research*, 24(164):1–81, 2023.
- [12] Nick Janssens and Johan Meyers. Towards real-time optimal control of wind farms using large-eddy simulations. *Wind Energy Science*, 9(1):65–95, 2024.
- [13] William B Johnson, Joram Lindenstrauss, et al. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.
- [14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

- [15] Jean Kossaifi, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, and Anima Anandkumar. Multi-grid tensorized fourier neural operator for high- resolution pdes. *Trans. Mach. Learn. Res.*, 2024.
- [16] Jean Kossaifi, Nikola Kovachki, Zongyi Li, David Pitt, Miguel Liu-Schiaffini, Robert Joseph George, Boris Bonev, Kamyar Azizzadenesheli, Julius Berner, Valentin Duruisseaux, and Anima Anandkumar. A library for learning neural operators. *arXiv preprint arXiv:2412.10354*, 2025.
- [17] Yuqi Li, Chuanguang Yang, Hansheng Zeng, Zeyu Dong, Zhulin An, Yongjun Xu, Yingli Tian, and Hao Wu. Frequency-aligned knowledge distillation for lightweight spatiotemporal forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7262–7272, 2025.
- [18] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.
- [19] Zongyi Li, Miguel Liu-Schiaffini, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Learning chaotic dynamics in dissipative systems. *Advances in Neural Information Processing Systems*, 35:16768–16781, 2022.
- [20] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024.
- [21] Jhe-Hao Lin, Yi Yao, Chan-Feng Hsu, Hong-Xia Xie, Hong-Han Shuai, and Wen-Huang Cheng. Perspective-aware teaching: Adapting knowledge for heterogeneous distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4178–4187, 2025.
- [22] Yuxuan Liu, Jingmin Sun, Xinjie He, Griffin Pinney, Zecheng Zhang, and Hayden Schaeffer. Prose-fd: A multimodal pde foundation model for learning multiple operators for forecasting fluid dynamics. *arXiv preprint arXiv:2409.09811*, 2024.
- [23] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12009–12019, 2022.
- [24] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 2022.
- [25] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017.
- [26] EN Lorenz. Deterministic nonperiodic flow. *J. Atmos. Sci.*, 20:130–141, 1963.
- [27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.
- [28] Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *Advances in Neural Information Processing Systems*, volume 37, 2024.
- [29] Michael McCabe, Payel Mukhopadhyay, Tanya Marwah, Bruno Regaldo-Saint Blancard, Francois Rozet, Cristiana Diaconu, Lucas Meyer, Kaze WK Wong, Hadi Sotoudeh, Alberto Bietti, et al. Walrus: A cross-domain foundation model for continuum dynamics. *arXiv preprint arXiv:2511.15684*, 2025.

- [30] Hossein Mobahi, Mehrdad Farajtabar, and Peter Bartlett. Self-distillation amplifies regularization in hilbert space. *Advances in Neural Information Processing Systems*, 33:3351–3361, 2020.
- [31] Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Stuart B Dalziel, Drummond B Fielding, et al. The well: a large-scale collection of diverse physics simulations for machine learning. *Advances in Neural Information Processing Systems*, 37:44989–45037, 2024.
- [32] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3967–3976, 2019.
- [33] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.
- [34] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In Yoshua Bengio and Yann LeCun (eds.), *International Conference on Learning Representations*, 2015.
- [35] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference*, pp. 234–241. Springer, 2015.
- [36] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *International conference on machine learning*, pp. 4470–4479. PMLR, 2018.
- [37] Yair Schiff, Zhong Yi Wan, Jeffrey B Parker, Stephan Hoyer, Volodymyr Kuleshov, Fei Sha, and Leonardo Zepeda-Núñez. Dyslim: Dynamics stable learning by invariant measure for chaotic systems. In *Forty-first International Conference on Machine Learning*, 2024.
- [38] Junhong Shen, Tanya Marwah, and Ameet Talwalkar. UPS: efficiently building foundation models for PDE solving via cross-modal adaptation. *Trans. Mach. Learn. Res.*, 2024.
- [39] Suraj Srinivas and François Fleuret. Knowledge transfer with jacobian matching. In *International conference on machine learning*, pp. 4723–4731. PMLR, 2018.
- [40] Makoto Takamoto, Francesco Alesiani, and Mathias Niepert. Learning neural pde solvers with parameter-guided channel attention. In *International Conference on Machine Learning*, pp. 33448–33467. PMLR, 2023.
- [41] Mingxuan Tian, Haochen Mu, Tao Liu, Mengjiao Li, Donghong Ding, and Jianping Zhao. Physics-informed machine learning-based real-time long-horizon temperature fields prediction in metallic additive manufacturing. *Communications Engineering*, 4(1):168, 2025.
- [42] Han Wan, Rui Zhang, and Hao Sun. Spectral-inspired neural operator for data-efficient pde simulation in physics-agnostic regimes. *arXiv preprint arXiv:2505.21573*, 2025.
- [43] Zhanhong Ye, Zining Liu, Bingyang Wu, Hongjie Jiang, Leheng Chen, Minyan Zhang, Xiang Huang, Qinghe Meng Zou, Hongsheng Liu, Bin Dong, et al. Pdeformer-2: A versatile foundation model for two-dimensional partial differential equations. *arXiv preprint arXiv:2507.15409*, 2025.
- [44] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3713–3722, 2019.
- [45] Rui Zhang, Qi Meng, Han Wan, Yang Liu, Zhi-Ming Ma, and Hao Sun. Omnifluids: Unified physics pre-trained modeling of fluid dynamics. *arXiv preprint arXiv:2506.10862*, 2025.

- [46] Maksim Zhdanov, David Ruhe, Maurice Weiler, Ana Lucic, Johannes Brandstetter, and Patrick Forré. Clifford-steerable convolutional neural networks. In *International Conference on Machine Learning*, pp. 61203–61228. PMLR, 2024.
- [47] Hang Zhou, Yuezhou Ma, Haixu Wu, Haowen Wang, and Mingsheng Long. Unisolver: Pde-conditional transformers towards universal neural PDE solvers. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR, 2025.

A Limitations

While TREX provides an effective framework for distilling PDE foundation models, it comes with several limitations. First, TREX relies on the quality of the fine-tuned teacher. Since the method samples from the teacher-induced rollout distribution, any systematic bias or instability in the teacher’s long-horizon dynamics may be transferred to the student. In particular, if the teacher converges to an incorrect or distorted attractor, TREX may reinforce this behavior. Although mixing teacher-generated data with ground-truth trajectories mitigates this issue, it does not fully eliminate it. Second, rollout extension is most useful when long trajectories continue to explore informative regions of state space. Its benefit may therefore be limited for systems that rapidly collapse to a trivial fixed point, exhibit strongly transient dynamics, or contain multiple disconnected long-time regimes that are not represented by the available trajectories. Moreover, the Gaussian perturbations used by TREX are not guaranteed to satisfy all physical constraints of the underlying PDE and should therefore be viewed as a state-space augmentation strategy rather than as physically valid perturbations.

B Broader Impact

This work aims to advance machine learning methods for scientific simulation and time-dependent PDEs by making foundation-model-based surrogates more efficient and easier to deploy. Faster surrogate models could reduce the computational cost of simulation workflows and make such tools more accessible in scientific and engineering applications. We do not expect broad negative societal impacts from the method itself.

C Compute Resources

The experiments were performed on a cluster with 10 NVIDIA GeForce RTX 4090 GPUs and on a larger compute cluster with 32 NVIDIA A100 GPUs and 32 NVIDIA L40 GPUs. Each individual training run used a single GPU. The reported inference-time and memory measurements for Poseidon experiments were collected for a batch size of 64, as described in the main text.

D Related Work

Knowledge distillation [4, 10] aims to provide the student with additional information from the teacher network. For classification tasks, the class logits can be transferred to the student [2, 10]. For general tasks, the student can be trained to have feature representations close to those of the teacher [34]. Relational KD [32] trains the student by comparing the relationship between the student’s feature representations of different inputs and the teacher’s, i.e., pushing the student to have the same distance between inputs as the teacher. Thus, the teacher and student do not need the same feature space representation. Similarly, Lin et al. [21] introduce a region-aware attention mechanism that reconstructs student patches into representations more closely aligned with the teacher’s view, even when the two models have substantially different architectures. Lastly, the student can also be trained to match the teacher’s derivatives. [39] train a student by matching the teacher’s Jacobian matrix. [1] apply a similar approach to distill a foundation model for molecules into a smaller student model by matching the teacher’s energy-prediction Hessian. In addition to providing additional information about the encountered training examples, it is also possible to artificially generate new training instances using the teacher model. For example, Lopes et al. [25] introduce data-free distillation, which aims to generate new samples that match the statistics of a dataset for which only those statistics are known.

Distillation has also been applied to PDEs or spatiotemporal predictions in general. [17] distill a special teacher architecture with branches for low- and high-frequency features into the student using separate losses for each branch. Wan et al. [42] propose SINO, which first learns a fine-step surrogate from a small number of trajectories and subsequently distills it into a coarse-step FNO using a larger teacher-generated dataset. The synthetic trajectories are generated from newly sampled initial conditions, whereas TREX targets settings in which no such initial-condition sampler is available and instead expands the available downstream trajectories through teacher rollouts. [45] present Omnifluids. Omnifluids first trains a high-resolution model on a PDE with varying parameters using a finite-difference loss for a known equation, similar to PINO [20]. The teacher is distilled into a

lower-resolution model with the same architecture by matching the teacher’s predictions on samples from the IC, assuming the teacher’s parameter distribution is known. In contrast, we consider general foundation models pretrained on large datasets, and do not assume knowledge of the PDE or input distributions.

PDE Foundation Models. In recent years, a number of PDE foundation models have been presented. Most use large vision transformers as their backbone [9, 22, 28, 29]. The earliest example of a PDE foundation model for nonlinear problems is MPP [28], which also uses axial attention to reduce the computational load. Since then, model and dataset sizes have been increasing. Poseidon [9] is based on a SwinV2 [23] vision transformer trained to predict the future state with the step size as a model input. Hence, it can be employed both for direct and autoregressive prediction. Walrus [29] is trained on The Well data [31] and can solve 2D and 3D tasks. While using a vision transformer, it also takes multiple time steps as inputs to infer the underlying physics from the observed trajectory segment. Zhou et al. [47] present Unisolver, which provides the vision transformer backbone with additional information about the PDE, such as the domain geometry or the physical parameters. Most notably, the equation symbols are embedded using an LLM. In contrast, PDEformer-2 [43] uses a graph-based transformer, and DPOT [6] is based on Fourier attention layers. [38] propose to use a pretrained LLM backbone to propagate in time. Additionally, they also embed additional inputs using an LLM similar to Unisolver.

Infusing Attractor Knowledge. Several authors have investigated explicitly including attractor dynamics for neural PDE solvers. Li et al. [19] include a loss term to enforce that the model always converges to the attractor on artificial data drawn uniformly from a fixed ball around zero. DySLIM [37] enforces that the learned model follows the invariant measure by minimizing the Maximum Mean Discrepancy between the data distribution and the learned attractor. [3] proposes a Poincaré Flow Neural Network, which is trained by dividing the data into the contracting and measure-invariant phases that can occur in a dissipative chaotic system. Depending on the phase, a loss is applied to enforce the contracting or invariant behavior.

E Datasets

We mainly use the datasets provided by [9]. All datasets, besides Kolmogorov Flow, contain 8 time steps, which end at $T = 0.7$ and use periodic boundary conditions. The spatial resolution is 128×128 for a grid of $[0, 1] \times [0, 1]$. The Poseidon problems have 120 validation and 240 test trajectories.

E.1 NS-PwC

The Navier-Stokes piecewise constants task uses the incompressible Navier-Stokes equations:

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} + \nabla p = \nu \Delta \mathbf{u}, \nabla \cdot \mathbf{u} = 0. \quad (2)$$

For NS-PwC, the initial state is given as a set of constants on a grid of 10 cells in each dimension:

$$\omega_0(x, y) = c_{i,j} \quad \text{for } (x, y) \in [x_{i-1}, x_i] \times [y_{j-1}, y_j], \quad i, j = 1, \dots, p. \quad (3)$$

with $x_i = y_i = \frac{i}{p}$, for $i = 0, 1, 2, \dots, p$ and the constant square values drawn from $c_{i,j} \sim \mathcal{U}_{[-1,1]}$. Here, ω_0 refers to the vorticity that can be transformed into the velocity in the x and y directions using the incompressibility condition.

E.2 NS-BB

In Fourier space, the Brownian bridge initial conditions are initialized as

$$W(x) = \sum_{|k|_\infty \leq N} \frac{1}{\|k\|_2^{3/2}} \sum_{m,n,\ell \in \{0,1\}} \alpha_k^{(mn\ell)} sc_m(x) sc_n(x) sc_\ell(x), \quad (4)$$

where

$$sc_i(x) = \begin{cases} \sin(x), & \text{for } i = 0, \\ \cos(x), & \text{for } i = 1 \end{cases}. \quad (5)$$

$W(x)$ does not directly give u_0 . Instead, it defines the state at $t=-0.5$, and the initial conditions are derived by evolving the Navier-Stokes equations until $t=0$. The random variables are sampled from $\alpha_k^{(mn\ell)} \sim \mathcal{U}_{[-1,1]}$.

E.3 NS-SVS

The next Navier-Stokes task uses Sinusoidal Vortex Sheet (SVS) initial conditions, which also use a vorticity formulation

$$\omega_0^\rho = \phi_\rho * \omega_0, \quad (6)$$

where ω_0 is defined as

$$\omega_0(x) = \delta(x - \Gamma) - \int_{\mathbb{T}^2} d\Gamma, \quad (7)$$

and

$$\phi_\rho(x) = \rho^{-2} \psi\left(\frac{\|x\|}{\rho}\right). \quad (8)$$

$\psi(r)$ is defined as

$$\psi(r) = \frac{80}{7\pi} \left[(r+1)_+^3 - 4 \left(r + \frac{1}{2}\right)_+^3 + 6r_+^3 - 4 \left(r - \frac{1}{2}\right)_+^3 + (r-1)_+^3 \right], \quad (9)$$

and Γ as

$$\Gamma = \left\{ (x, y) \in \mathbb{T}^2 \left| y = \frac{1}{2} + 0.2 \sin(2\pi x) + \sum_{i=1}^p \alpha_i \sin(2\pi(x + \beta_i)) \right. \right\}. \quad (10)$$

The variable parameters are drawn from

$$\alpha_i \sim \mathcal{U}_{[0, 0.003125]}, \quad \beta_i \sim \mathcal{U}_{[0, 1]}, \quad (11)$$

and ρ and p are fixed to $\rho = \frac{5}{128}$ and $p = 10$.

E.4 CE-RPUI

The Compressible Euler PDE is given as

$$u_t + \nabla \cdot F(u) = 0, \quad (12)$$

where u and $F(u)$ are defined as

$$u = \begin{bmatrix} \rho \\ \rho \mathbf{v} \\ E \end{bmatrix}, \quad F(u) = \begin{bmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + p \mathbf{I} \\ (E + p) \mathbf{v} \end{bmatrix}. \quad (13)$$

For the CE-RPUI problem, the four channels are drawn as constant values for each region of the input grid. The input division sets $D_{i,j}$ are computed using sinusoidal functions $\sigma_x(x, y)$ and $\sigma_y(x, y)$, which are defined as

$$\sigma_x(x, y) = \sum_{i,j=1}^p \alpha_{x,i,j} \sin(2\pi(i + 2p^2)x + (j + 2p^2)y + \beta_{x,i,j}), \quad (14)$$

and similarly

$$\sigma_y(x, y) = \sum_{i,j=1}^p \alpha_{y,i,j} \sin(2\pi(i + 2p^2)x + (j + 2p^2)y + \beta_{y,i,j}). \quad (15)$$

The coefficients $\alpha_{k,i,j}$ $\beta_{k,i,j}$ are drawn from uniform distributions:

$$\alpha_{k,i,j} \sim \mathcal{U}[-0.01, 0.01], \quad \beta_{k,i,j} \sim \mathcal{U}[0, 1]. \quad (16)$$

Afterward, the input space regions are defined as

$$D_{i,j} = \left\{ (x, y) \in \mathbb{T}^2 \left| \begin{array}{l} x_{\min} \leq \{x + \sigma_x(x, y) + 1\} < x_{\max}, \\ y_{\min} \leq \{y + \sigma_y(x, y) + 1\} < y_{\max} \end{array} \right. \right\}, \quad (17)$$

where $\{x\} := x - \lfloor |x| \rfloor \operatorname{sgn}(x)$ and

$$x_{\min} = \frac{i}{p+1}, \quad x_{\max} = \frac{i+1}{p+1}, \quad y_{\min} = \frac{j}{p+1}, \quad y_{\max} = \frac{j+1}{p+1}. \quad (18)$$

In the end, the initial condition is defined using the regions $D_{i,j}$

$$(\rho, v_x, v_y, p)|_{t=0} = (\rho_{i,j}, v_{i,j}^x, v_{i,j}^y, p_{i,j}) \quad \text{in } D_{i,j}, \quad (19)$$

where the region-wise parameters are drawn from

$$\rho_{i,j} \sim \mathcal{U}[1, 3], \quad v_{i,j}^x \sim \mathcal{U}[-10, 10], \quad v_{i,j}^y \sim \mathcal{U}[-10, 10], \quad p_{i,j} \sim \mathcal{U}[5, 7]. \quad (20)$$

E.5 Kolmogorov Flow

For the ablations, we use the Kolmogorov flow dataset [19] at a Reynolds number (Re) of 5000. The PDE can be described as

$$\frac{\partial u}{\partial t} = -u \cdot \nabla u - \nabla p + \frac{1}{Re} \Delta u + \sin(4y)\hat{x}, \quad (x, y, t) \in [0, 2\pi] \times [0, 2\pi] \times [0, \infty) \quad (21)$$

$$\nabla \cdot u = 0, \quad (x, y, t) \in [0, 2\pi] \times [0, 2\pi] \times [0, \infty) \quad (22)$$

$$u(\cdot, 0) = u_0, \quad (x, y) \in [0, 2\pi] \times [0, 2\pi]. \quad (23)$$

We transform the PDE into velocity form so that it can be included in Walrus’ channel mapping. The resolution is 128x128. The trajectories are 501 time steps long, where the first 100 are discarded. We use 48 trajectories for validation and test each. For training, only a single trajectory is provided.

F Model and Training Details

All student models are trained to predict the difference, i.e., $\hat{\mathbf{u}}^{t+1} = \mathcal{S}(\mathbf{u}^t) = \mathcal{M}(\mathbf{u}^t) + \mathbf{u}^t$. The learning rate is set to 10^{-3} initially and scaled down to 10^{-6} using cosine annealing. We use the standard Adam optimizer [14] and a batch size of 32. The experiments were performed on NVIDIA GeForce RTX 4090 GPUs (one GPU per run). For the main results, we report the test error for the model checkpoint with the best error on the validation set for the ground-truth only and relational KD baselines, since we encountered overfitting for these methods. For TREX and IC-KD, we report the last model checkpoint to be fair to the Teacher baseline, which also did not use early-stopping. We use the original training loss from Poseidon over the batch $\mathcal{B}$:

$$L_{\text{train}}^1(\theta) = \frac{1}{N_{\text{QOI}}} \sum_{\text{QOI}} \frac{\sum_{b \in \mathcal{B}} \sum_t \sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} |\hat{\mathbf{u}}_{\theta, b}(t, \mathbf{x}, c_{\text{QOI}}) - \mathbf{u}_b(t, \mathbf{x}, c_{\text{QOI}})|}{\sum_{b \in \mathcal{B}} \sum_t \sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} |\mathbf{u}_b(t, \mathbf{x}, c_{\text{QOI}})| + \varepsilon_{L_1}}.$$

TFNO. We use the Tucker factorized FNO [15] version with the implementation provided by [16]. The factorization reduces the number of parameters with a compression factor of 0.1. The network uses 32 modes per dimension, 32 hidden channels, three layers, and a GELU [8] activation function.

U-Net. For the student model ablations, we use the modern U-Net [35] as provided by [5]. The U-Net has 4 levels of resolution, increasing the number of channels per level by 1, 2, 2, and 4, starting at 16 hidden channels after the embedding, resulting in 9,161,332 parameters. For U-Net, we found it advantageous to train with a 5-step error for the model ablation experiment.

FNO. We also use a traditional FNO [18] without factorization for the model ablations. The model uses the same number of modes and channels as the TFNO, resulting in 5,042,836 parameters. The FNO uses 4 layers.

Poseidon. As the teacher, we use the Poseidon-L model [9]. Poseidon-L was pretrained on six different datasets containing Navier-Stokes and compressible Euler equations with different sets of initial conditions. The model uses a SwinV2 transformer as its backbone, a multi-resolution vision transformer. The architecture operates on a latent representation with embedding dimension $C = 192$. For each hierarchical level i , the model employs $t_i = 8$ blocks, resulting in a total of 629M trainable parameters. The network consists of $L = 4$ levels, connected via $L - 1$ downsampling and upsampling operations to progressively refine spatial resolution. The number of attention heads per level is defined as:

$$[h_1, h_2, h_3, h_4] = [3, 6, 12, 24],$$

allowing increased representation capacity at coarser scales. In addition, each level incorporates two ConvNeXt layers [24]. The model operates on input patches of size $p = 4$ and a window size of $M = 16$. We report the original median relative L^1 error metric from Poseidon [9], which normalizes the L^1 loss for each quantity of interest (QOI) frame-wise:

$$L_{\text{QOI}}^1(\mathbf{u}, \hat{\mathbf{u}}) = \frac{\sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} |\hat{\mathbf{u}}(t_{\text{last}}, \mathbf{x}, c_{\text{QOI}}) - \mathbf{u}(t_{\text{last}}, \mathbf{x}, c_{\text{QOI}})|}{\sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} |\mathbf{u}(t_{\text{last}}, \mathbf{x}, c_{\text{QOI}})|}. \quad (24)$$

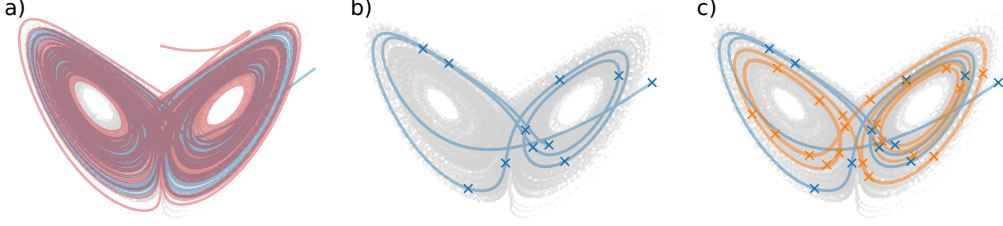


Figure 3: Long rollouts can increase state-space coverage in dynamical systems. Two trajectories of the Lorenz system [26] visit similar regions of state space over time (a). Starting from a short observed trajectory (b), extending the rollout generates additional states from the long-time trajectory distribution (c).

Training is performed using the AdamW optimizer [27] with cosine learning rate decay. The main learning rate is initialized as $\eta_b = 5 \cdot 10^{-5}$. A weight decay of 10^{-6} is applied for regularization. The batch size is set to 16, and the model is trained for 200 epochs without early stopping. Gradient norms are clipped at a maximum value of 5 to ensure numerical stability.

Walrus. For Walrus [29] fine-tuning, we used only 50k sample presentations to the model (batch size 4). We used global normalization and followed the original training procedure otherwise. For the distillation, we started 50 rollouts from different random sub-sequences in the trajectory of length 100. We found noise during Walrus rollout harmful, likely since adding noise to a multi-frame input does not just create a different state, but also suggests different physics to the model if it learned to infer it from its inputs. More importantly, it was crucial to deactivate the jitter added in Walrus inference, as this creates stochastic predictions and hence noisy labels. Walrus uses the VRMSE metric:

$$\text{VRMSE}_{t_1:t_2}(\hat{\mathbf{u}}, \mathbf{u}) = \frac{1}{t_2 - t_1 + 1} \sum_{t=t_1}^{t_2} \sqrt{\frac{\langle |\hat{\mathbf{u}}_t - \mathbf{u}_t|^2 \rangle}{\langle |\mathbf{u}_t - \bar{\mathbf{u}}_t|^2 \rangle + \varepsilon}}. \quad (25)$$

We use the normalized mean absolute error as the loss function

$$\mathcal{L}_W(\mathbf{u}, \hat{\mathbf{u}}) = \frac{1}{T} \sum_{t=0}^{T-1} |\text{Norm}_\Delta(\hat{\mathbf{u}}^{t+1} - \mathbf{u}^t) - \text{Norm}_\Delta(\mathbf{u}^{t+1} - \mathbf{u}^t)|_1.$$

The normalization is performed with the global mean and standard deviation of the transition differences.

G Knowledge Distillation Details

In this section, we provide additional results on the KD techniques used in this paper.

TREX For TREX, we use 100 rollout steps starting from the initial condition by default. The noise is applied after 10 steps, with a scaling factor of $\sigma = 1$. The TREX loss weight is set to 1 as well. The noisy rollouts are regenerated every 1000 epochs.

The ergodic Lorenz attractor example in Fig. 3 further illustrates the motivation behind our method: although trajectories start from different initial conditions, their long rollouts repeatedly visit a shared region of the state space. TREX exploits the same principle for PDE foundation models by turning a small number of ground-truth trajectories into a much broader set of teacher-labeled states that eventually cover the same space.

IC-KD For the IC-KD baseline, we use a buffer of teacher-generated trajectories that we regenerate every 10 epochs, with the size of the buffer determined by the number of ground truth trajectories available. The teacher-generated rollouts are then also divided into sub-trajectories for n-step training.

Note that for the Brownian bridge task, the initial conditions are computationally intensive to compute because they are evolved from an earlier state.

$$\mathcal{L}_{\text{IC}} = \mathbb{E}_{\mathbf{u}_0 \sim p_0} \left[\frac{1}{T} \sum_{t=0}^{T-1} \ell(\mathcal{S}(\mathbf{u}^t), \mathcal{T}(\mathbf{u}^t)) \right], \quad \mathbf{u}^{t+1} = \mathcal{T}(\mathbf{u}^t)$$

Relational KD Relational KD [32] aims to make the distance between an input pair i, j from a mini-batch in the student feature space close to the distance in the feature space:

$$\mathcal{L}_{\text{RKD}} = \sum_{i < j} \ell_\delta \left(d(\mathbf{t}_i, \mathbf{t}_j), d(\mathbf{s}_i, \mathbf{s}_j) \right), \quad (26)$$

where $\mathbf{t}$ and $\mathbf{s}$ are the teacher and student features, respectively. The loss is the Huber distance

$$\ell_\delta(x, y) = \begin{cases} \frac{1}{2}(x - y)^2 & \text{if } |x - y| \leq 1 \\ |x - y| - \frac{1}{2} & \text{otherwise} \end{cases}. \quad (27)$$

Since the feature spaces of the spatial neural network are very large, we use Gaussian sketching, which uses random Gaussian matrices R to reduce the input features to a lower dimension for the teacher

$$\mathbf{t}_i = \mathbf{R}_T \tilde{\mathbf{t}}_i, \quad (\mathbf{R}_T)_{ab} \sim \mathcal{N}\left(0, \frac{1}{k}\right), \quad \mathbf{R}_T \in \mathbb{R}^{k \times d_T}, \quad (28)$$

and equivalently, the student model

$$\mathbf{s}_i = \mathbf{R}_S \tilde{\mathbf{s}}_i, \quad (\mathbf{R}_S)_{ab} \sim \mathcal{N}\left(0, \frac{1}{k}\right), \quad \mathbf{R}_S \in \mathbb{R}^{k \times d_S}. \quad (29)$$

It can be shown that the error in the distances $d(\mathbf{t}_i, \mathbf{t}_j)$ introduced by the Gaussian sketch is bounded [11, 13]. We use $k = 512$ in all experiments.

The loss $\mathcal{L}_{\text{RKD}}$ is added to the main, ground truth loss with a scaling factor of 0.001. We use the last-layer features for both the student and the teacher.

H Further Results

Tab. 2 shows the main results for the median relative L^1 metric with the 95% confidence interval of the mean. For completeness, we also show the relative L^1 error for all time steps (Tab. 3). Additionally, we report the training durations in Tab. 4. Fig. 4 shows a full example rollout for the NS-SVS task and Fig. 5 shows the error growth for all 4 considered tasks.

Table 2: Median rel. L^1 metric of the *last* time step. Entries show the mean with its 95% confidence interval calculated over three seeds.

Algorithm	$N = 4$	$N = 8$	$N = 16$	$N = 32$
CE-RPUI				
TREX	0.421 ± 0.015	0.295 ± 0.182	0.197 ± 0.038	0.144 ± 0.010
IC-KD	0.440 ± 0.077	0.304 ± 0.167	0.223 ± 0.105	0.145 ± 0.006
Relational KD	0.608 ± 0.099	0.446 ± 0.068	0.296 ± 0.031	0.196 ± 0.020
Only GT	0.608 ± 0.086	0.458 ± 0.092	0.296 ± 0.011	0.199 ± 0.018
Teacher	0.446 ± 0.074	0.310 ± 0.157	0.223 ± 0.108	0.143 ± 0.009
NS-SVS				
TREX	0.043 ± 0.037	0.021 ± 0.005	0.014 ± 0.001	0.011 ± 0.002
IC-KD	0.045 ± 0.032	0.025 ± 0.005	0.018 ± 0.003	0.015 ± 0.004
Relational KD	0.178 ± 0.049	0.050 ± 0.020	0.021 ± 0.004	0.011 ± 0.001
Only GT	0.171 ± 0.053	0.060 ± 0.021	0.023 ± 0.005	0.011 ± 0.001
Teacher	0.043 ± 0.030	0.024 ± 0.005	0.018 ± 0.005	0.015 ± 0.003
NS-PwC				
TREX	0.056 ± 0.003	0.042 ± 0.017	0.032 ± 0.012	0.021 ± 0.003
IC-KD	0.067 ± 0.012	0.053 ± 0.039	0.042 ± 0.038	0.029 ± 0.007
Relational KD	0.175 ± 0.014	0.059 ± 0.011	0.027 ± 0.003	0.013 ± 0.001
Only GT	0.282 ± 0.081	0.068 ± 0.011	0.029 ± 0.002	0.013 ± 0.001
Teacher	0.070 ± 0.011	0.055 ± 0.040	0.044 ± 0.035	0.031 ± 0.005
NS-BB				
TREX	0.036 ± 0.007	0.026 ± 0.007	0.023 ± 0.006	0.016 ± 0.008
IC-KD	0.041 ± 0.010	0.034 ± 0.004	0.031 ± 0.005	0.023 ± 0.009
Relational KD	0.111 ± 0.022	0.034 ± 0.007	0.015 ± 0.001	0.007 ± 0.002
Only GT	0.178 ± 0.077	0.041 ± 0.015	0.015 ± 0.003	0.007 ± 0.001
Teacher	0.046 ± 0.010	0.039 ± 0.005	0.033 ± 0.003	0.025 ± 0.008

Table 3: Median rel. L^1 metric over *all* timesteps. Entries show the mean with its 95% confidence interval calculated over three seeds.

Algorithm	$N = 4$	$N = 8$	$N = 16$	$N = 32$
CE-RPUI				
TREX	0.277 ± 0.027	0.182 ± 0.065	0.127 ± 0.019	0.095 ± 0.008
IC-KD	0.281 ± 0.035	0.184 ± 0.080	0.135 ± 0.055	0.093 ± 0.005
Relational KD	0.471 ± 0.074	0.316 ± 0.042	0.197 ± 0.025	0.129 ± 0.014
Only GT	0.466 ± 0.067	0.327 ± 0.049	0.201 ± 0.011	0.130 ± 0.005
Teacher	0.282 ± 0.038	0.185 ± 0.078	0.129 ± 0.055	0.084 ± 0.005
NS-SVS				
TREX	0.020 ± 0.014	0.009 ± 0.002	0.006 ± 0.001	0.004 ± 0.001
IC-KD	0.020 ± 0.009	0.011 ± 0.002	0.008 ± 0.001	0.006 ± 0.001
Relational KD	0.070 ± 0.023	0.019 ± 0.009	0.008 ± 0.002	0.004 ± 0.001
Only GT	0.068 ± 0.024	0.022 ± 0.009	0.008 ± 0.002	0.004 ± 0.001
Teacher	0.020 ± 0.008	0.011 ± 0.002	0.008 ± 0.001	0.006 ± 0.001
NS-PwC				
TREX	0.032 ± 0.002	0.024 ± 0.007	0.019 ± 0.004	0.014 ± 0.002
IC-KD	0.037 ± 0.009	0.028 ± 0.016	0.023 ± 0.014	0.017 ± 0.003
Relational KD	0.105 ± 0.011	0.037 ± 0.007	0.018 ± 0.002	0.010 ± 0.001
Only GT	0.172 ± 0.048	0.043 ± 0.008	0.019 ± 0.002	0.010 ± 0.001
Teacher	0.039 ± 0.011	0.029 ± 0.016	0.024 ± 0.014	0.018 ± 0.002
NS-BB				
TREX	0.022 ± 0.005	0.016 ± 0.005	0.013 ± 0.002	0.009 ± 0.003
IC-KD	0.023 ± 0.006	0.019 ± 0.004	0.017 ± 0.001	0.012 ± 0.004
Relational KD	0.067 ± 0.012	0.022 ± 0.004	0.010 ± 0.001	0.005 ± 0.001
Only GT	0.108 ± 0.050	0.026 ± 0.009	0.010 ± 0.002	0.005 ± 0.001
Teacher	0.026 ± 0.007	0.022 ± 0.002	0.018 ± 0.001	0.014 ± 0.004

Table 4: Average training duration in minutes on NS-PwC.

Method	$N = 4$	$N = 8$	$N = 16$	$N = 32$
TREX	47.1 ± 1.9	99.8 ± 3.2	147.4 ± 1.4	291.4 ± 4.6
IC-KD	85.2 ± 2.3	117.7 ± 4.5	193.5 ± 1.4	338.7 ± 6.1
Relational KD	38.2 ± 1.4	72.9 ± 2.7	143.1 ± 3.8	280.2 ± 9.8
Only Ground Truth	37.3 ± 0.4	70.2 ± 0.4	138.4 ± 4.3	271.5 ± 7.8

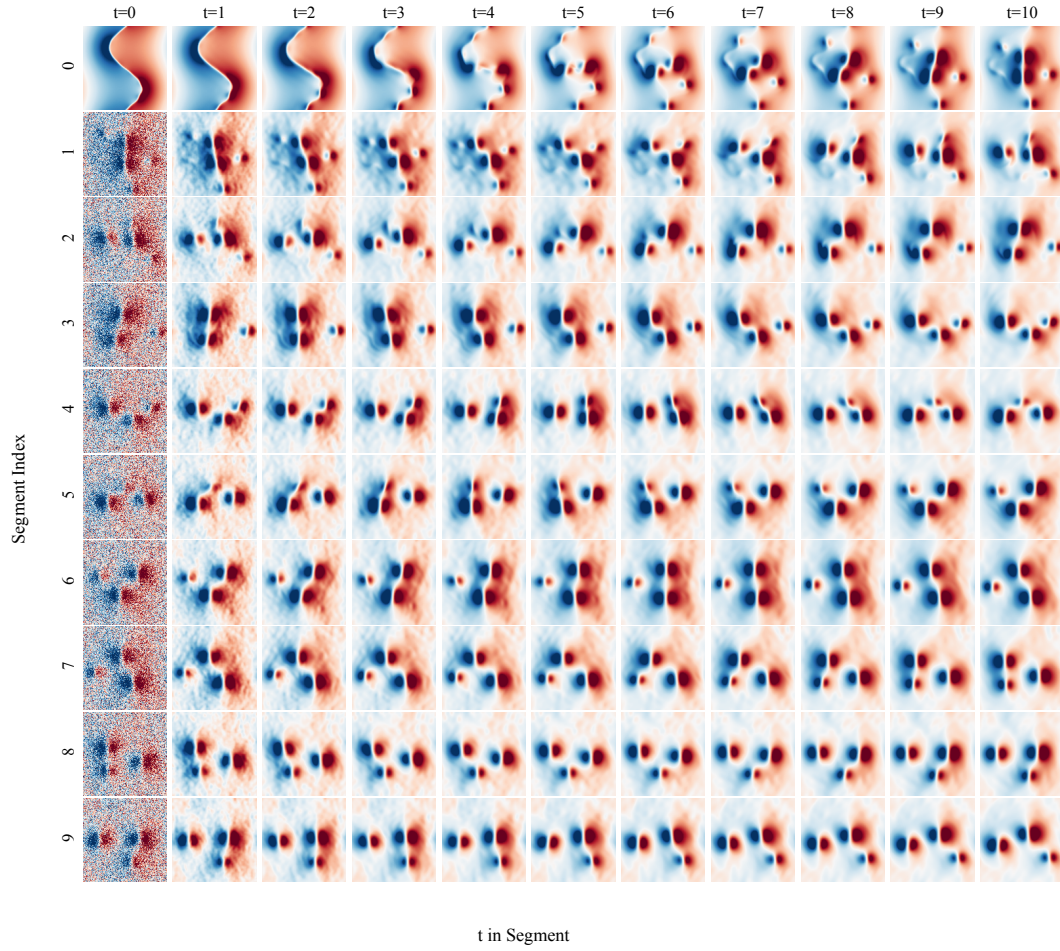


Figure 4: Example trajectory (velocity-x channel) generated using the teacher for $\sigma = 1$ on NS-SVS. After each sub-trajectory (row), the noise is applied.

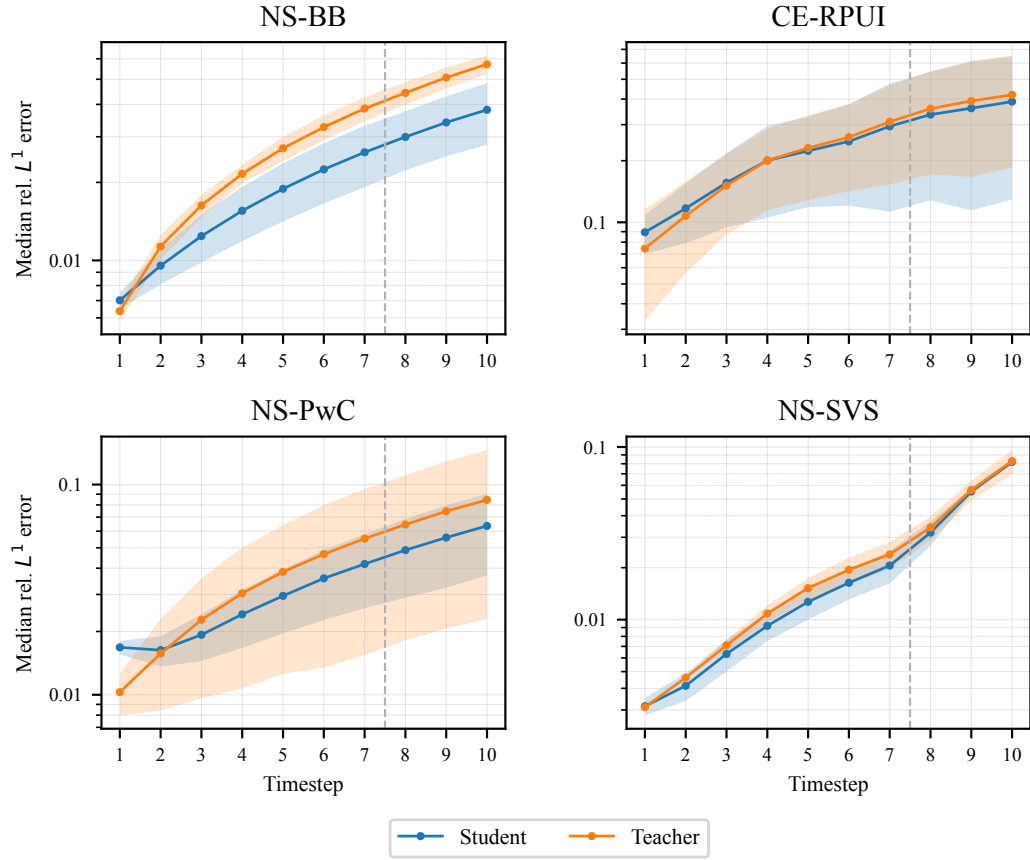


Figure 5: Error over time for the student trained with TREX (8 trajectories). The shaded area shows the 95% confidence interval of the mean (3 seeds). The dashed line shows the time horizon seen during training.

H.1 Ablations

To investigate why TREX can generate a student that outperforms its teacher, we conduct an ablation study. To this end, we select the NS-PwC-4 case and fully remove the ground-truth trajectories, training the student only on teacher-generated data (Tab. 5). Even without the ground truth data, TREX still outperforms its teacher. Removing noise injections during the rollout reduces the gap, but not fully. The phenomenon that distillation can improve student performance has been observed in the general KD literature [30, 32]. It can be explained by regularization effects, which can even occur when distilling the same model into the same architecture [30, 44]. Next to the core algorithm

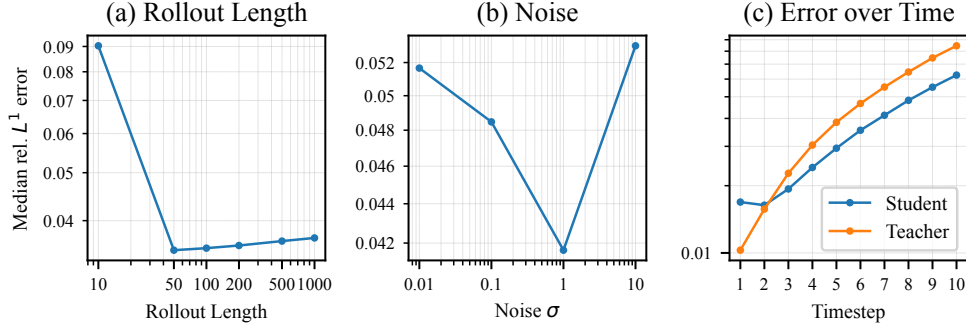


Figure 6: Ablation study on NS-SVS (four trajectories). The effect of the rollout is relatively constant over a wide range of hyperparameters (a). Injecting noise also reduces the error, particularly at relatively high noise amplitudes (b). In (c), the error over time is shown for NS-PwC (8 trajectories). Time steps $t > 7$ show temporal extrapolation beyond the training time horizon. While the TREX-trained student has a higher one-step error than the teacher, the student has a more stable rollout.

choices, we also investigate the influence of the included hyperparameters. Fig. 6a) shows the effect of the rollout length. The algorithm is quite robust to the choice of this hyperparameter. The performance only decreases slowly for very long rollouts. The effect of the strength of the injected noise is shown in Fig. 6b). Interestingly, choosing the noise quite high at a level of one time the channel-wise standard deviation s_c produces the best results.

H.1.1 Student Model Ablation

We also provide an experiment with other student architectures. We use a modern UNet [5] and a standard FNO [18]. As shown in Fig. 7, TREX is also able to let other student architectures match the teacher’s accuracy for all considered numbers of trajectories. While the TFNO was able to improve over the teacher using only ground truth for a higher number of trajectories in the main experiments (Fig. 2), this was not the case for the UNet or even a standard FNO, and was only possible through distillation.

H.1.2 Teacher Model Ablation

We also experiment with other teacher models. Walrus [29] is a larger vision transformer (1.3B parameters) that uses the last six states as context for next state prediction. This setup is crucially

Table 5: Ablation study on NS-PwC (four trajectories). Shown are the effects of removing the noise injection during the teacher rollout, as well as removing ground truth trajectories fully from the student training.

Variant	Median rel. L^1
TREX	0.056 ± 0.003
w/o GT	0.060 ± 0.002
w/o GT & w/o Noise	0.064 ± 0.006
Teacher	0.070 ± 0.011

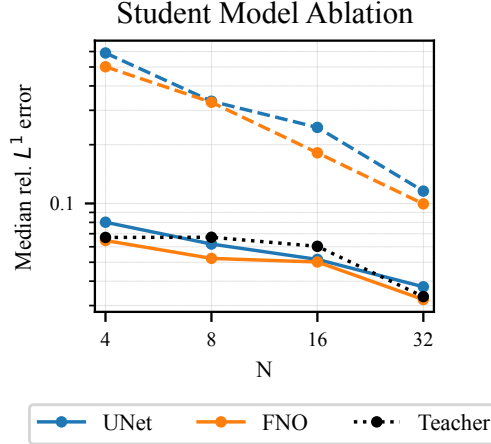


Figure 7: Ablation on the student architecture (NS-PwC). Shown are the results for training with TREX (full) and only ground-truth data (dashed).

Table 6: Distillation of Walrus [29] on the Kolmogorov Flow dataset. TREX improves the long-rollout accuracy while increasing the inference speed.

	VRMSE _{1-Step}	VRMSE _{$T \in [1:60]$}	Inference Time (s)
Walrus	0.012	0.547	21.425
TREX	0.016	0.498	0.414
Only GT	0.050	0.766	0.401

different since it does not allow IC-KD, as we would need to run the simulator to get the initial sequence to start the rollout. To also go beyond the rather short trajectories provided by the Poseidon datasets, we test distillation on the 2D Kolmogorov Flow dataset from [19] with a Reynolds number of 5000. The trajectories have 501 time steps, where the first 100 are discarded [19]. For the student, we select a higher capacity TFNO with 64 modes and a feature dimension of 80, trained for only 10k epochs with batch size 64. Similar to the teacher, the student is using six input steps as the context. For the ground truth only baseline, we found a single step as input to work better. We fine-tune Walrus on a single trajectory and report the VRMSE metric (Eq. (25)) for different rollout intervals. The student trained with TREX can also successfully distill the multi-step teacher foundation model (Tab. 6).

H.2 Self-Distillation

We also perform a self-distillation experiment to further examine the influence of the teacher pre-training. We use the TFNO trained only on ground-truth samples as the teacher for distillation into a freshly initialized model of the same architecture. The results in Tab. 7 show that, while self-distillation has a small positive regularizing effect, the main advantage comes from distilling the pretrained, higher-capacity foundation model.

Table 7: Self-distillation experiment using TREX on CE-RPUI (4 trajectories).

Method	Median rel. L^1
Only GT	0.608
TREX (self-distillation)	0.585
TREX (from Poseidon)	0.421

Table 8: Translation equivariance results. Left: median relative L^1 equivariance error, computed using the unperturbed predictions as ground truth. Right: final-step relative L^1 errors on original and shifted OOD rollouts.

Dataset	Equivariance error		Final-step rollout error			
	Teacher	Student	Poseidon	Poseidon shift	FNO	FNO shift
NS-BB	0.026 ± 0.013	0.000 ± 0.001	0.0651 ± 0.0286	0.0676 ± 0.0203	0.0504 ± 0.0080	0.0504 ± 0.0080
CE-RPUI	0.101 ± 0.006	0.001 ± 0.001	0.4189 ± 0.0767	0.4108 ± 0.0856	0.3549 ± 0.0344	0.3549 ± 0.0344
NS-PwC	0.023 ± 0.006	0.000 ± 0.001	0.1095 ± 0.0420	0.1041 ± 0.0385	0.0744 ± 0.0206	0.0744 ± 0.0206
NS-SVS	0.110 ± 0.005	0.000 ± 0.001	0.0820 ± 0.0083	0.2398 ± 0.0086	0.0755 ± 0.0072	0.0755 ± 0.0072

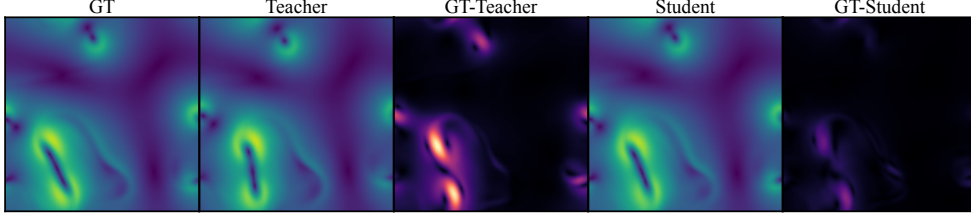


Figure 8: Final-step shifted-rollout comparison on NS-SVS. The teacher shows a visibly larger error with respect to the shifted ground truth (GT), while the student prediction remains much closer and the corresponding difference is substantially smaller.

H.3 Equivariance

To investigate the effect of the explicit equivariance built into the student architecture, we test how the models behave under spatial shifts on the periodic grid. We apply random 2D rolls to the input fields, which corresponds to the translation symmetry of the underlying periodic domain. An equivariant model should produce the same prediction up to this same roll. We evaluate this in two ways: first as a direct equivariance error, and second through shifted autoregressive rollouts.

For the direct equivariance metric, let $a = (d_x, d_y)$ denote a sampled 2D periodic translation, and let τ_a be the corresponding shift operator. We compare the prediction on the shifted input with the shifted prediction on the original input, $\|f(\tau_a x) - \tau_a f(x)\|_1$, using the same median relative L^1 normalization as in the main evaluation. As shown in Tab. 8, the student has substantially lower equivariance error than the teacher across all datasets. This is expected from the student architecture, but it is still important: distillation transfers the teacher’s predictive behavior into a model class that enforces a symmetry not explicitly respected by the teacher.

We further evaluate the effect of equivariance in autoregressive rollouts for temporal extrapolation by applying the same spatial shift to the initial condition and comparing the final prediction to the shifted ground truth. The rollout results in Tab. 8 show that the shifted rollout error of the student remains essentially unchanged, while the teacher can degrade substantially, most clearly on NS-SVS. Since NS-SVS also has the largest teacher equivariance error, we show a qualitative example in Fig. 8. In this case, the teacher produces visibly worse shifted rollouts, while the student remains consistent with the translated dynamics. We include the entire rollout trajectory in Appendix Fig. 9. These results suggest that equivariance can be a useful property of the distilled student, especially in autoregressive simulations where small symmetry errors may accumulate over long rollouts. More broadly, it is encouraging that TREX can use a non-equivariant teacher to train a compact student that preserves the teacher’s predictive behavior while enforcing a physically meaningful symmetry of the downstream task.

As a complement to the final-step comparison in Fig. 8, Fig. 9 shows the complete shifted rollout on NS-SVS. On the unshifted rollout, the teacher initially appears slightly closer to the ground truth than the student. After applying the spatial shift, however, the teacher prediction quickly degrades, starting around steps 3/4, while the student remains consistent throughout the rollout.

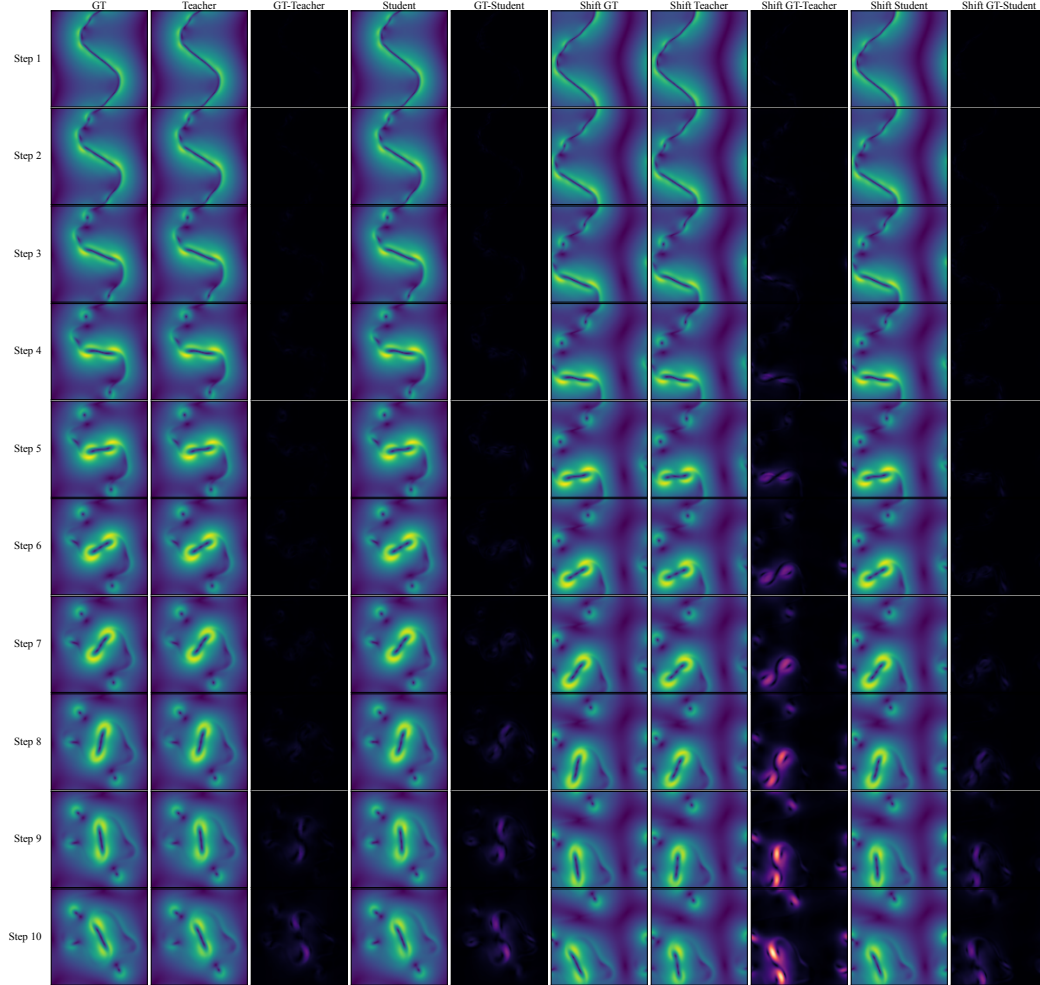


Figure 9: Full shifted-rollout comparison on NS-SVS. Each row shows one autoregressive step. The teacher accumulates larger errors under the spatial shift, while the equivariant student remains consistent with the shifted ground truth throughout the rollout.