MuPS: Learning Multiple Physical Solution Branches from Unlabeled Outcomes

Anonymous authors
Paper under double-blind review

Abstract

Learning-based physical simulation is typically built around an input–output map: given a physical condition, a model predicts a solution, or samples one from a conditional distribution. This formulation becomes inadequate when the underlying physical problem is intrinsically multi-solution, where the correct prediction target is a condition-dependent set of admissible solutions. In such settings, a single predictor may average incompatible solutions, while a probabilistic model may require many samples to reveal distinct or rare outcomes. We identify this as a fundamental limitation of current learning-based physics models and formulate the problem as learning a set-valued solution map $a \mapsto \mathcal{S}(a)$ from branch-unlabeled input–solution observations. We introduce MuPS, which augments existing physics-learning architectures with a small collection of explicit solution hypotheses and condition-dependent activation. The hypotheses provide direct access to distinct solution modes, while activation determines which hypotheses are admissible for each condition, allowing the represented solution-set cardinality to vary across the input domain. We demonstrate that this formulation applies to Fourier neural operators, neural fields, and conditional diffusion models. Across three physical and synthetic benchmarks and three model families, MuPS substantially improves best-of-K prediction accuracy on multi-solution problems, often by orders of magnitude, while providing more sample-efficient access to distinct solutions. It remains effective under strongly imbalanced solution observations and can retain comparable performance on a standard single-solution problem.

1 Introduction

Learning-based physical simulation has largely been developed around learning an input–output operator: a physical condition is mapped to a predicted solution. Neural operators (Li et al., 2021; 2023; Zhou et al., 2026), neural fields (Raissi et al., 2019; Chen et al., 2023a; Kairanda et al., 2024; Han et al., 2026), and related learned simulators have achieved impressive results under this formulation. Probabilistic approaches, including diffusion-based simulators (Shysheya et al., 2024; Huang et al., 2024; Holzschuh et al., 2026), extend it by learning a conditional distribution over possible outcomes.

Unfortunately, many physical problems have multiple solutions. As illustrated in Figure 1, the same physical condition can admit multiple valid solutions (Thompson & Hunt, 1973). In this case, the quantity of interest is not a single-valued operator $a \mapsto u$, but a *set-valued* map

$$a \mapsto \mathcal{S}(a),$$

where $\mathcal{S}(a)$ denotes the admissible solution set. Its cardinality may vary with a, and its members may correspond to qualitatively different physical states. A deterministic model can collapse these states into a single prediction, while a probabilistic model represents them only implicitly and may require repeated sampling to expose rare modes. Neither formulation directly provides efficient access to the admissible solution set.

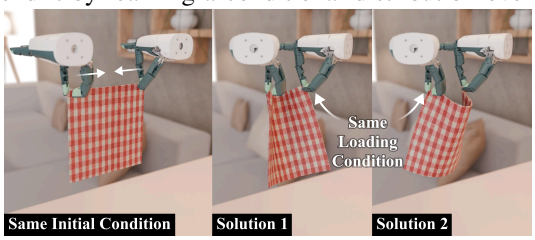

Figure 1: **A single physical condition may admit multiple valid solutions.** Buckling is a canonical example, where the same loading condition can admit distinct equilibrium configurations.

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.

We therefore study how to extend learning-based physical simulation from single-valued prediction to condition-dependent solution-set recovery. Training provides branch-unlabeled input–solution observations $\mathcal{D} = (a_i, u_i)_{i=1}^N$, potentially with multiple observations for the same condition. These samples reveal members of the underlying solution set, but do not provide branch identities or correspondences across conditions. The goal is to recover a representation of $\mathcal{S}(a)$ that exposes distinct admissible solutions directly and can adapt its effective cardinality to the physical condition.

Recent work has begun to address multiplicity explicitly in physical systems. HomPINNs (Zheng et al., 2024) use homotopy and governing-equation constraints to recover multiple solutions from unlabeled observations, while NINO (Hao et al., 2024) incorporates nonlinear-solver structure near bifurcation points. Bifurcation Models (Jore & Liu, 2026) formulate scientific prediction as learning a set-valued solution map. These approaches highlight the importance of multi-solution physical prediction, but rely on governing equations, solver structure, physical objectives, or other problem-specific information. We instead ask whether set-valued physical prediction can be learned directly from input–solution data and integrated into existing physics-learning architectures.

Our approach builds on ideas from multiple-hypothesis learning (Rupprecht et al., 2017), mixture models (Bishop, 1994), and conditional generative models (Sohn et al., 2015). In particular, multiple-hypothesis learning provides a natural mechanism for training explicit predictions to specialize through best-match assignment. We use this mechanism as a building block for set-valued physical prediction: a small collection of learned hypotheses provides direct, budgeted access to distinct solution modes, while a condition-dependent activation model determines which hypotheses are admissible for each input. This turns a fixed hypothesis budget into a condition-dependent solution set and applies uniformly to Fourier neural operators, neural fields, and conditional diffusion models.

Across three benchmarks and three model families, our approach substantially improves best-of-K prediction accuracy on multi-solution problems, often by orders of magnitude, improves solution-set coverage and sample-efficient access to distinct solutions, and can retain comparable performance on a standard single-solution problem.

In summary, we identify a limitation of common data-driven physics-learning formulations: they predict a single solution, or sample outcomes from a conditional distribution, even when the underlying physical problem admits a condition-dependent set of valid solutions. We formulate this as direct condition-to-solution-set learning and, to our knowledge, introduce the first data-driven framework for learning condition-dependent sets of physical solutions from branch-unlabeled observations, without governing equations, solver structure, physical objectives, or branch labels. We demonstrate the approach on benchmarks spanning nonlinear PDEs, deformable mechanics, and continuum robotics, and across three model families, showing improved best-of-K accuracy, solution-set coverage, and access to rare solutions under fixed prediction budgets.

2 Related Work

Learning-based physical simulation. Learning-based physical simulation spans learned dynamics and hybrid simulators (Pfaff et al., 2021; Brandstetter et al., 2022; Lippe et al., 2023; Li et al., 2024a; 2026; Wang et al., 2026; Viswanath et al., 2026), neural operators (Li et al., 2021; 2023; Tran et al., 2023; Lingsch et al., 2024; Zhou et al., 2026), and physics-informed models (Raissi et al., 2019; Wang et al., 2021; Li et al., 2024b). Neural fields and implicit neural representations (Chen & Zhang, 2019; Park et al., 2019; Sitzmann et al., 2020) provide continuous spatial representations for PDEs, mechanics, and fluid simulation (Yang et al., 2022; Chen et al., 2023a;b; Serrano et al., 2023; Deng et al., 2023; Jain et al., 2024; Kairanda et al., 2024; Tao et al., 2024; Chang et al., 2023; 2025a;b; Liu et al., 2025; Modi et al., 2024). Diffusion models provide a flexible framework for generative modeling through iterative denoising or score estimation (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021). Recent work has applied diffusion-based generative modeling to PDE simulation, forecasting, inverse problems, and probabilistic physical surrogates (Shysheya et al., 2024; Huang et al., 2024; Holzschuh et al., 2026). These works primarily develop more expressive or efficient simulation architectures; our formulation is complementary and instead addresses how such models can recover a condition-dependent set of admissible solutions from incomplete observations.

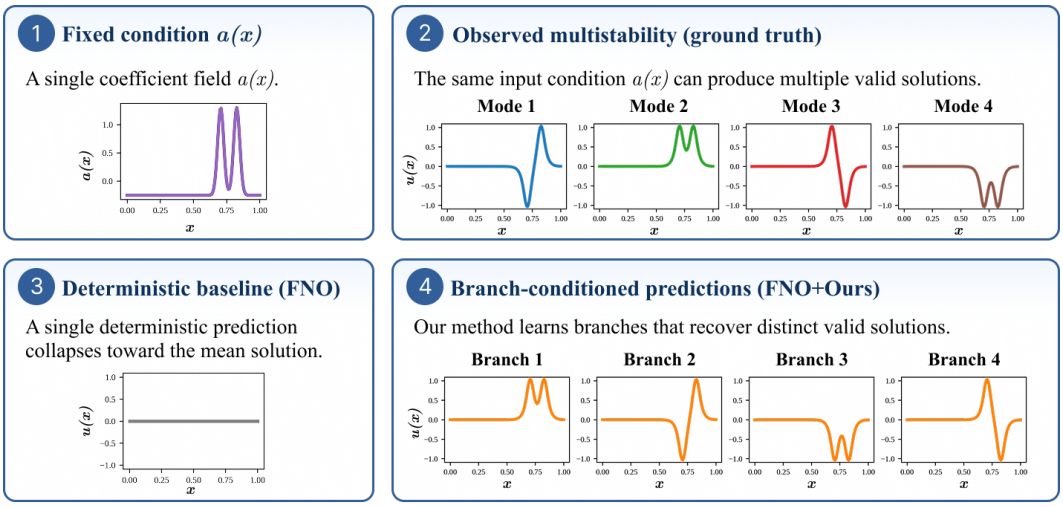

Figure 2: **Multi-solution prediction in a multistable reaction–diffusion system.** For a fixed coefficient field $a(x)$, repeated simulations of the reaction–diffusion system converge to multiple distinct valid steady states. A deterministic FNO produces a single prediction that collapses toward the mean solution and therefore does not represent the individual modes. In contrast, our branch-conditioned formulation learns multiple predictions that recover the distinct valid solutions associated with the same input condition.

Learning multiple physical solutions. Recent work addresses multiplicity in nonlinear physical systems. HomPINNs (Zheng et al., 2024) combine physics-informed neural networks with homotopy continuation to recover multiple solutions, including from unlabeled observations, while NINO (Hao et al., 2024) learns a Newton-based nonlinear PDE solver near bifurcation points. Both rely on governing equations or solver structure. Most closely related, Bifurcation Models (Jore & Liu, 2026) learn set-valued solution maps through stable equilibria of weight-tied dynamics, but branch discovery relies on a physical objective such as an energy functional. Neural ODEs have also recovered bifurcation structure from trajectories conditioned on a known bifurcation parameter (van Tegelen et al., 2025). In contrast, we use only branch-unlabeled input–solution pairs and directly learn candidate solutions together with their condition-dependent admissibility.

Multimodal prediction. One-to-many prediction has long been studied through mixture models (Bishop, 1994), conditional latent-variable models (Sohn et al., 2015; Lavda et al., 2020), and multiple-hypothesis or best-of-K objectives (Guzmán-Rivera et al., 2012; Lee et al., 2016; Rupprecht et al., 2017). Our approach builds on best-match specialization Rupprecht et al. (2017); Firman et al. (2018) to recover and efficiently access an admissible physical solution set $\mathcal{S}(a)$ whose structure and cardinality may vary with the condition.

On its own, best-match specialization on physical datasets produces redundant branches (see Figures 5 and 13). To address this shortcoming, our work further learns which candidates are admissible for each condition through a novel form of activation, which we coin *condition-dependent activation*. We compare against Gaussian- and mixture-prior CVAEs, MDNs, and multiple-hypothesis prediction in Section 5.4.

3 Multi-Solution Learning for Conditional Physics Models

Our formulation treats multi-solution physics learning as condition-dependent solution-set recovery and is largely agnostic to the underlying model architecture. We assume a conditional physics model that maps a condition $a \in \mathcal{A}$ to a solution $\hat{u} \in \mathcal{U}$, possibly through additional stochastic variables for generative models. This covers FNOs, coordinate-based neural fields, and conditional diffusion models. We augment the model with K learned solution hypotheses, defining a fixed budget of candidate solutions for each condition. Using best-match training, different hypotheses

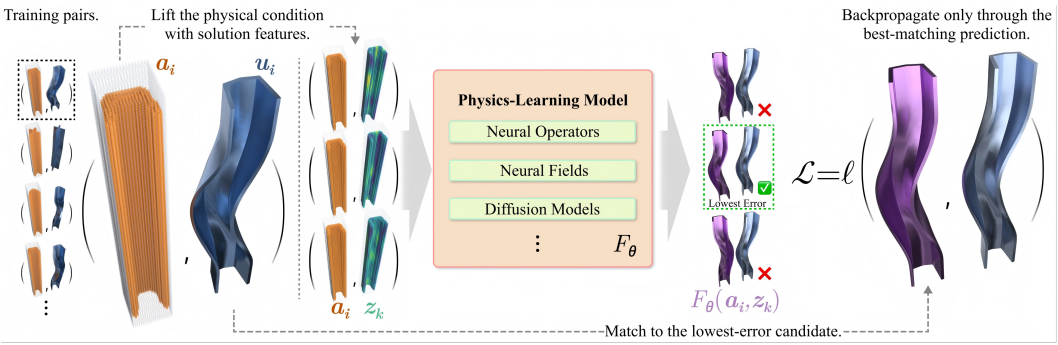

Figure 3: **Learning solution hypotheses.** Learned features produce multiple candidates, and each observation updates only its best-matching prediction.

can specialize to different solution modes without branch labels. A separate condition-dependent activation model then determines which candidates are admissible for a particular a, allowing the represented solution-set cardinality to vary across conditions. We next describe the hypothesis-learning and activation objectives, followed by their instantiations for deterministic and diffusion-based physics models.

3.1 Learning Solution Hypotheses

We represent the unknown solution set using K learned solution features $\{z_k\}_{k=1}^K$ and a shared physics model F_θ,

$$\hat{u}_k(a) = F_\theta(a, z_k), \qquad k = 1, \dots, K. \tag{1}$$

The features have no predefined physical identities. Since each training pair (a_i, u_i) reveals only one valid solution, we route it to the candidate with the smallest architecture-specific loss,

$$k_i^* = \arg\min_k \ell(\hat{u}_k(a_i), u_i), \qquad \mathcal{L}_{\text{fit}} = \frac{1}{N}\sum_{i=1}^{N} \ell(\hat{u}_{k_i^*}(a_i), u_i). \tag{2}$$

Here ℓ is a reconstruction loss for deterministic models and a denoising loss for diffusion models.

To discourage hypothesis collapse, we add a pairwise hinge separation loss over all candidates,

$$\mathcal{L}_{\text{sep}} = \frac{2}{NK(K-1)} \sum_{i=1}^{N} \sum_{k<k'} \max(0,\, m - \|\hat{u}_k(a_i) - \hat{u}_{k'}(a_i)\|_2)^q, \tag{3}$$

with $q = 2$ in our experiments, and optimize

$$\mathcal{L}_{\text{hyp}} = \mathcal{L}_{\text{fit}} + \lambda_{\text{sep}} \mathcal{L}_{\text{sep}}. \tag{4}$$

The margin m is calibrated from the architecture- and dataset-specific prediction-distance scale rather than fixed globally. This prevents the separation term from becoming either inactive or overly dominant when output magnitudes differ across models and datasets.

3.2 Condition-Dependent Solution Activation

The K hypotheses form an upper bound on the admissible solution set, but not all need be valid for every condition. We therefore predict independent activation probabilities

$$\mathbf{p}(a) = \sigma(G_\phi(a)) = [p_1(a), \dots, p_K(a)], \tag{5}$$

where G_ϕ receives the same condition field as the underlying physics model.

We train the activation model while keeping the candidate model fixed. Let d_{ik} denote the discrepancy between observation u_i and candidate k. A soft coverage objective encourages at least one active candidate to explain each observation,

$$\mathcal{L}_{\text{cov}} = -\frac{1}{N}\sum_{i=1}^{N} \log \sum_{k=1}^{K} p_k(a_i) \exp\left(-\frac{d_{ik}}{T}\right). \tag{6}$$

To avoid retaining redundant candidates, we additionally estimate a target solution count M_i from repeated observations at the same and nearby conditions, and penalize only activation beyond this budget,

$$\mathcal{L}_{\text{sparse}} = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{k=1}^{K} p_k(a_i) - M_i\right]_+ . \quad (7)$$

Together with a diversity penalty on simultaneously active, near-duplicate predictions, the activation objective is

$$\mathcal{L}_{\text{act}} = \mathcal{L}_{\text{cov}} + \lambda_{\text{sparse}}\mathcal{L}_{\text{sparse}} + \lambda_{\text{div}}\mathcal{L}_{\text{div}}. \quad (8)$$

At inference time, candidates with $p_k(a) \geq 0.5$ are retained, yielding a condition-dependent solution set. Full implementation details are provided in the appendix.

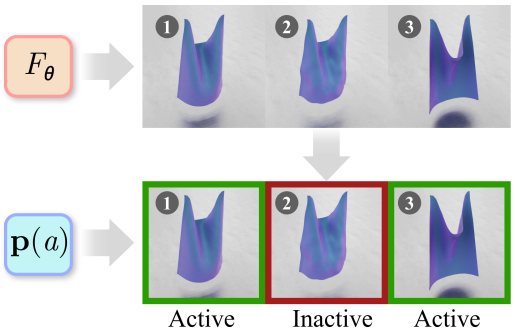

Figure 4: **Condition-dependent activation.** Active candidates are selected for each input.

4 Instantiations in Physics-Learning Models

Our formulation requires only that the underlying simulator can be conditioned on an additional learned feature. We instantiate it with three different physics-learning architectures.

Fourier neural operators. For FNOs, the learned feature is a smooth trainable spatial field concatenated with the physical input,

$$\hat{u}_k = F_\theta([a, z_k]). \quad (9)$$

Candidate matching uses the relative H^1 loss

$$\ell^{\text{FNO}}(\hat{u}_k, u) = \frac{\|\hat{u}_k - u\|_{H^1}}{\|u\|_{H^1} + \epsilon}, \quad (10)$$

while activation uses solution-space MSE, $d_{ik} = D^{-1}\|\hat{u}_k(a_i) - u_i\|_2^2$.

Coordinate-based neural fields. For neural fields, z_k is shared across all spatial queries,

$$\hat{u}_k(\mathbf{x}) = F_\theta(\mathbf{x}, a, z_k). \quad (11)$$

Both candidate matching and activation use the spatially averaged L_2 error,

$$\ell^{\text{Field}}(\hat{u}_k, u) = d_{ik} = \frac{1}{M}\sum_{j=1}^{M}\|\hat{u}_k(\mathbf{x}_j; a_i) - u_i(\mathbf{x}_j)\|_2^2. \quad (12)$$

Diffusion-based simulation. For conditional diffusion, a and z_k remain fixed throughout denoising,

$$u_{t-1} = D_\theta(u_t, t, a, z_k). \quad (13)$$

Different z_k therefore condition distinct generative processes while retaining the stochasticity of the base model. Candidate matching uses the standard noise-prediction objective,

$$\ell^{\text{Diffusion}} = \frac{1}{D}\|\hat{\epsilon}_k - \epsilon\|_2^2. \quad (14)$$

For activation, discrepancies are instead computed from sampled reconstructed solutions of the frozen diffusion hypotheses, as detailed in Appendix A.3.

5 Experiments and Results

5.1 Experiments

Reaction–diffusion system. We first consider the one-dimensional reaction–diffusion system

$$\partial_t u = \varepsilon\partial_{xx}u + a(x)u - u^3, \quad (15)$$

where $a(x)$ contains two localized positive regions. For each $a(x)$, hidden random perturbations yield multiple final states. We use 32,768 training and 4,096 evaluation samples per split.

Table 1: **Multi-solution handling consistently improves prediction accuracy across different physics-learning frameworks and datasets.** We report test-set mean squared error (MSE, ↓). Deterministic baselines are evaluated using their single prediction. For our method, we report the minimum MSE over the K candidate solutions. For diffusion baselines, we draw K independent samples and likewise report the minimum MSE, matching the number of predictions used by our method.

	Reaction–Diffusion (MSE ↓)	Cloth (MSE ↓)	Robot (MSE ↓)
Neural Field	1.25×10^{-1}	1.81×10^{-2}	2.52×10^{-4}
Neural Field + Ours	$\mathbf{6.00 \times 10^{-6}}$	$\mathbf{8.58 \times 10^{-4}}$	$\mathbf{1.80 \times 10^{-7}}$
FNO	1.25×10^{-1}	2.78×10^{-2}	4.29×10^{-4}
FNO + Ours	$\mathbf{4.38 \times 10^{-3}}$	$\mathbf{2.96 \times 10^{-4}}$	$\mathbf{1.40 \times 10^{-7}}$
Cond. Diffusion	6.96×10^{-2}	1.53×10^{-2}	9.60×10^{-5}
Cond. Diffusion + Ours	$\mathbf{6.57 \times 10^{-5}}$	$\mathbf{2.37 \times 10^{-3}}$	$\mathbf{1.60 \times 10^{-6}}$

Table 2: **Effect of condition-dependent solution activation.** We report test-set symmetric set MSE (↓) between the observed solutions and the predicted candidate set. "Without Activation" evaluates all learned candidate branches, while "With Activation" retains only branches selected by the activation model. Bold indicates the lower error for each backbone and dataset.

	Reaction–Diffusion (MSE ↓)	Cloth (MSE ↓)	Robot (MSE ↓)
Neural Field w/o Activation	$\mathbf{4.22 \times 10^{-6}}$	5.05×10^{-3}	8.28×10^{-4}
Neural Field + Activation	$\mathbf{4.22 \times 10^{-6}}$	$\mathbf{7.11 \times 10^{-4}}$	$\mathbf{1.69 \times 10^{-4}}$
FNO w/o Activation	$\mathbf{2.89 \times 10^{-3}}$	3.94×10^{-3}	2.02×10^{-4}
FNO + Activation	$\mathbf{2.89 \times 10^{-3}}$	$\mathbf{6.07 \times 10^{-4}}$	$\mathbf{1.69 \times 10^{-4}}$
Cond. Diffusion w/o Activation	$\mathbf{1.16 \times 10^{-2}}$	3.64×10^{-1}	9.48×10^{-2}
Cond. Diffusion + Activation	2.21×10^{-2}	$\mathbf{4.12 \times 10^{-2}}$	$\mathbf{6.52 \times 10^{-2}}$

Cloth simulation. We evaluate nonlinear cloth simulations with buckling-induced multistability. For each physical condition, we simulate the same thin volumetric domain using multiple tetrahedral discretizations, treating discretization differences as unobserved perturbations. We consider the setting of two pinned regions moved inward, producing two dominant buckling modes.

Tendon-driven continuum robot. We evaluate on the planar tendon-driven continuum robot of Peyron & Burgner-Kahrs (2024). For a range of tendon tensions T_1, we solve the static Kirchhoff-rod equilibrium equations using shooting, continuation, and multi-start initialization to recover disconnected stable branches. The same actuation condition can therefore admit multiple configurations, with solution multiplicity changing across bifurcations.

Table 1 compares neural fields (Kairanda et al., 2024), FNOs (Li et al., 2021), and conditional diffusion models (Shysheya et al., 2024) with the same backbones augmented by our multi-solution formulation. Across all three benchmarks, our formulation reduces best-of-K test error for all three model families, with particularly large gains for deterministic models. Conditional diffusion already represents multiple outcomes through stochastic sampling, but branch-conditioned predictions further improve accuracy under the same prediction budget.

Figure 2 illustrates this behavior qualitatively. Deterministic baselines collapse multiple valid outcomes into a single prediction, whereas our formulation recovers distinct solution modes. Diffusion can access multiple solutions through sampling, while our branch-conditioned representation provides more direct and sample-efficient access, as discussed in Section A.4.

5.2 Condition-Dependent Solution Activation

To evaluate the condition-dependent solution activations, we compare the set of observed solutions with the subset of candidate predictions retained by the activation model. For a condition a, let $\mathcal{Y}(a) = \{u_j\}_{j=1}^{M}$ denote the observed solutions and $\widehat{\mathcal{U}}_{\text{act}}(a) = \{\hat{u}_k(a) : p_k(a) \geq \tau_{\text{act}}\}$ denote the activated candidate predictions. We measure their agreement using a symmetric nearest-neighbor

Table 3: **Effect of condition-dependent activation on branch-count recovery.** We report the mean absolute error (MAE, ↓) between the predicted and reference number of solution branches, together with the exact-match rate (↑), i.e., the percentage of test conditions for which the predicted branch count is exactly correct. "Without Activation" retains all K learned candidate branches, whereas "With Activation" retains only branches selected by the activation model.

		Mean True Count	MAE w/o Act. ↓	MAE w/ Act. ↓	Exact w/o Act. ↑	Exact w/ Act. ↑
Reaction–Diffusion	FNO	4.00	0.00	**0.00**	100.0%	**100.0%**
Reaction–Diffusion	Neural Field	4.00	0.00	**0.00**	100.0%	**100.0%**
Reaction–Diffusion	Cond. Diffusion	4.00	2.00	**0.00**	0.0%	**100.0%**
Cloth	FNO	2.00	2.00	**0.00**	0.0%	**100.0%**
Cloth	Neural Field	2.00	2.00	**0.00**	0.0%	**100.0%**
Cloth	Cond. Diffusion	2.00	2.00	**0.00**	0.0%	**100.0%**
Robot	FNO	1.33	2.67	**0.20**	0.0%	**88.3%**
Robot	Neural Field	1.33	2.67	**0.20**	0.0%	**88.3%**
Robot	Cond. Diffusion	1.33	2.67	**0.38**	0.0%	**73.3%**

set distance,

$$E_{\mathrm{cov}}(a) = \frac{1}{|\mathcal{Y}(a)|} \sum_{u \in \mathcal{Y}(a)} \min_{\hat{u} \in \widehat{\mathcal{U}}_{\mathrm{act}}(a)} d(u, \hat{u}), \quad E_{\mathrm{extra}}(a) = \frac{1}{|\widehat{\mathcal{U}}_{\mathrm{act}}(a)|} \sum_{\hat{u} \in \widehat{\mathcal{U}}_{\mathrm{act}}(a)} \min_{u \in \mathcal{Y}(a)} d(\hat{u}, u).$$

Their symmetric average, $E_{\mathrm{set}}(a) = \frac{1}{2}(E_{\mathrm{cov}}(a) + E_{\mathrm{extra}}(a))$, gives the final set error, where $d(\cdot, \cdot)$ is the per-solution MSE. The coverage term penalizes observed solutions not represented by any active prediction, while the excess-branch term penalizes active predictions not corresponding to an observed solution. This yields a Chamfer-style distance between the observed and activated solution sets.

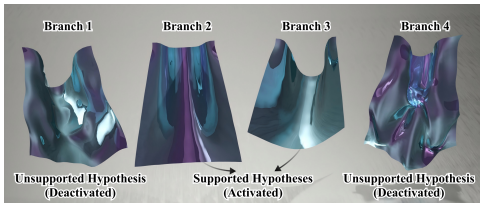

Figure 5: **Example solution hypotheses and their activation.** We visualize the deformations represented by all learned branches for one physical condition. The activation model retains hypotheses corresponding to supported solution modes and suppresses unsupported candidates.

Table 2 isolates the contribution of condition-dependent activation after multi-solution candidate learning. Thus, "Without Activation" is trained with our hypothesis-specialization objective; the comparison measures the additional benefit of reconstructing the admissible subset from those learned candidates. In many of the evaluated datasets, activation substantially reduces the set error for cloth and robot examples. These gains therefore cannot be attributed simply to learning multiple solutions: they show that producing diverse candidate solutions alone is insufficient when the number of admissible solutions varies with the physical condition. Figure 5 provides a qualitative example corresponding to the set-error improvements reported in Table 2. It illustrates the role of the activation model intuitively: the multi-solution predictor first produces a fixed pool of candidate deformations, while activation retains the hypotheses that correspond to admissible solution modes for the current condition and suppresses the remaining candidates.

Activation improves the result by removing redundant or unsupported hypotheses while preserving candidates needed to cover the observed solutions. For the neural-field and FNO models on reaction–diffusion, the error remains unchanged because the candidate budget K already matches the solution multiplicity, leaving essentially no excess candidates to remove. The benefit is larger when K is chosen conservatively to accommodate the maximum multiplicity across the dataset: multi-solution learning provides a sufficiently large candidate pool, while activation converts this fixed pool into a smaller condition-dependent solution set. Additional qualitative results are provided in Appendix A.6.1.

5.3 Branch-count accuracy.

We additionally evaluate whether activation recovers the correct *number* of solutions at each condition, independently of the reconstruction accuracy of individual candidates. For evaluation only,

we manually annotate the reference solution multiplicity $M_{\text{true}}(a)$ for each test condition from the available physical outcomes. These annotations are never used during training of either the solution hypotheses or the activation model. The predicted count is

$$M_{\text{pred}}(a) = |\{k : p_k(a) \geq \tau_{\text{act}}\}| \,. \tag{16}$$

using $\tau_{\text{act}} = 0.5$. We report the mean absolute count error and the fraction of test conditions for which $M_{\text{pred}}(a) = M_{\text{true}}(a)$ exactly. As a baseline, "without activation" always retains all K learned candidates, isolating the additional contribution of condition-dependent set reconstruction beyond multi-solution candidate learning. Table 3 shows that condition-dependent activation substantially improves branch-count recovery beyond multi-solution candidate learning alone.

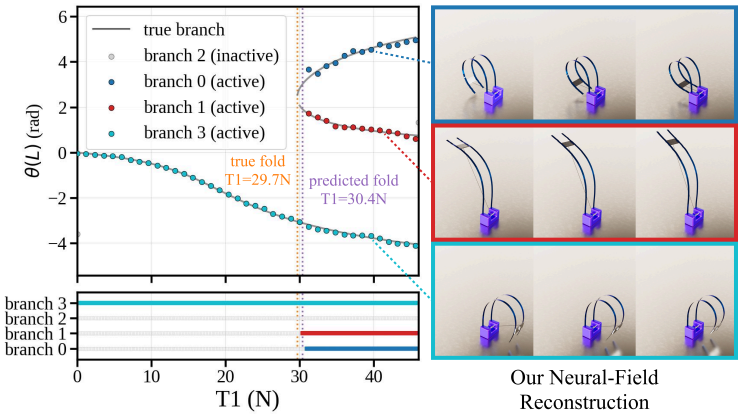

Figure 6: **Condition-dependent activation recovers the bifurcation structure of a tendon-driven continuum robot.** Using a neural-field backbone, our reconstructed solution branches and predicted activations closely match the reference stability analysis, with a predicted fold at $T_1 = 30.4\,\text{N}$ versus the reported $29.7\,\text{N}$.

Beyond recovering the correct solution count, the learned activations also capture meaningful changes in the underlying physical solution structure. Figure 6 demonstrates this on the tendon-driven continuum robot of Peyron & Burgner-Kahrs (2024). Using a neural-field backbone, we reproduce their bifurcation experiment by varying the tendon tension T_1 and predicting both candidate configurations and their condition-dependent activations. The activated branches closely follow the solution structure reported by the stability analysis: our predicted transition occurs at $T_1 = 30.4, \text{N}$, close to the reference fold bifurcation at $29.7, \text{N}$. The same transition is also recovered by the other backbones, with conditional diffusion predicting $27.5, \text{N}$ and FNO $30.7, \text{N}$. Branches that are not admissible under a given actuation condition are suppressed, leaving the corresponding valid configurations. Together with the branch-count results, this suggests that activation recovers not only a smaller candidate set, but also the condition-dependent multiplicity and availability of distinct physical solutions.

5.4 Connection to Multimodal Prediction

Our formulation is closely connected to classical multimodal prediction methods. We therefore compare our method with a Gaussian-prior CVAE (Sohn et al., 2015), a mixture-prior CVAE (Sohn et al., 2015; Lavda et al., 2020), and a mixture density network (MDN) (Bishop, 1994) on the bifurcation example in Figure 7, following the bifurcation setup of Peyron & Burgner-Kahrs (2024, Figure 6). As shown in the top row, while deterministic regression collapses the multi-valued mapping to a single prediction, all three multimodal baselines recover the multiple outcomes to varying degrees. This supports our broader motivation that techniques developed for multimodal prediction can help address the single-valued assumption common in physics-learning pipelines. Our method differs in representing the solutions as explicitly addressable branches with condition-dependent activation. We additionally compare alternative hypothesis parameterizations inspired by prior multiple-hypothesis methods in Appendix A.8 (Table 10).

We further evaluate how this representation affects access to the learned solution set under increasingly imbalanced training data. For stochastic baselines, a query corresponds to an additional sample from the learned conditional distribution; for MDN and our method, individual components or branches can instead be queried directly. Across all imbalance levels, Our method reaches its accuracy floor and near-complete solution-set coverage once its three learned branches have been queried, whereas sampling-based methods improve more gradually with additional samples. Stratified sampling improves the mixture-prior CVAE by explicitly visiting different mixture components, further illustrating the benefit of structured access to multimodal predictions. MDN also benefits

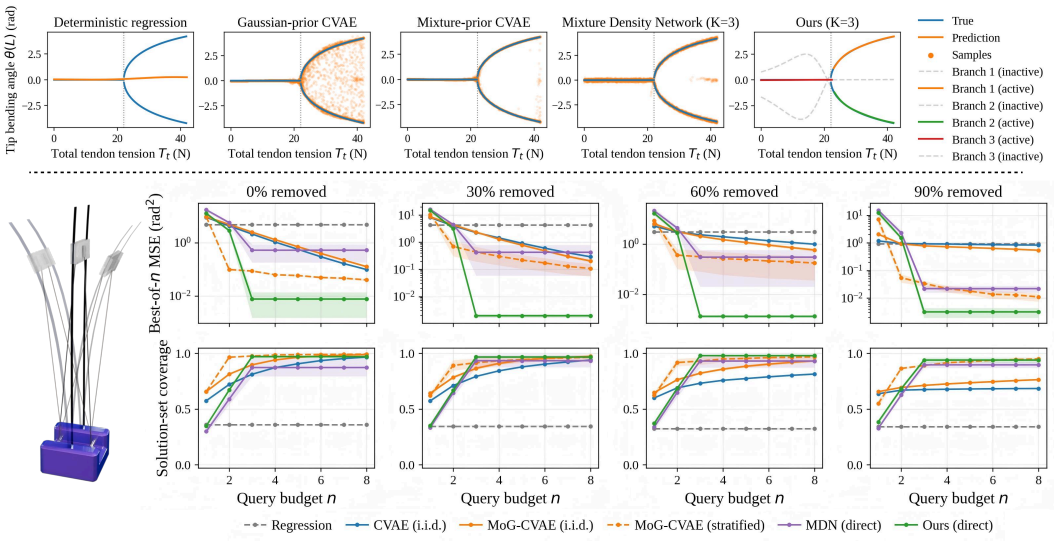

Figure 7: **Multimodal prediction and solution-set recovery under rare-branch imbalance.** Top: deterministic regression averages the multi-valued mapping, while multimodal baselines and our method recover multiple outcomes; our method represents them as explicit condition-dependent branches. Bottom: best-of-n error and stable-solution coverage as one branch is progressively removed from the training data. Stochastic methods use n samples, while MDN and our method directly query their components or branches. Our method reaches near-complete coverage after three branch queries and remains robust even when the rare branch accounts for only 5% of post-bifurcation training examples. Curves show mean $\pm$ standard error over 10 seeds.

from direct component queries, but exhibits substantially larger variation across training seeds as the rare branch becomes underrepresented. Together, these results suggest that multimodal prediction provides a useful foundation for multi-solution physics learning, while explicit branch specialization offers reliable and sample-efficient access to distinct physical solutions.

6 Additional Results.

The appendix provides further evaluation and analysis of the proposed formulation. We report matched-budget sampling efficiency for conditional diffusion (Section A.4), branch-aware inverse inference on the continuum robot (Section A.5), additional qualitative results across all three benchmarks (Section A.6), compatibility with a standard single-solution task (Section A.7), and ablations on the hypothesis budget, separation regularization, hypothesis parameterization, solution-feature dimension, and activation regularization (Section A.8). We also provide a simplified analysis of deterministic averaging and finite-budget coverage under probabilistic sampling A.9.

7 Conclusion and Limitations

We study multi-solution physics learning as condition-dependent solution-set recovery from branch-unlabeled observations. MuPS combines learned solution hypotheses with condition-dependent activation and applies across several physics-learning backbones. A fixed hypothesis budget provides a compact, directly accessible representation of the solution set; empirically, MuPS recovers distinct modes across all three benchmarks, adapts the active set across conditions, and remains robust when solution observations are strongly imbalanced. Because MuPS learns from observed outcomes alone, recovery is necessarily limited to solution structure supported by the available data unless additional physical assumptions are introduced. The current formulation allows branch identities to specialize locally rather than explicitly enforcing global cross-condition correspondence. Future work could build on the demonstrated branch recovery and activation behavior using adaptive hypotheses and stronger cross-condition consistency.

AI USE STATEMENT

Generative AI tools were used to assist with portions of method implementation and experimental code, debugging, feedback on experimental design, mathematical exposition and derivations, literature discovery, and manuscript drafting and editing. The research problem and core methodology, substantial portions of the evaluation design, simulation data generation, initial core implementations, and interpretation of results were carried out by the authors. AI-assisted outputs were used as intermediate research aids. The authors take responsibility for the final manuscript and reported results.

REFERENCES

Christopher M. Bishop. Mixture density networks. Technical Report NCRG/94/004, Aston University, 1994.

Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural pde solvers. In *International Conference on Learning Representations*, 2022.

Yue Chang, Peter Yichen Chen, Zhecheng Wang, Maurizio M. Chiaramonte, Kevin Carlberg, and Eitan Grinspun. LiCROM: Linear-subspace continuous reduced order modeling with neural fields. In *SIGGRAPH Asia 2023 Conference Papers*, pp. 1–12. Association for Computing Machinery, 2023. doi: 10.1145/3610548.3618158.

Yue Chang, Otman Benchekroun, Maurizio M. Chiaramonte, Peter Yichen Chen, and Eitan Grinspun. Shape space spectra. *ACM Transactions on Graphics*, 44(4):1–16, 2025a. doi: 10.1145/3731148.

Yue Chang, Mengfei Liu, Zhecheng Wang, Peter Yichen Chen, and Eitan Grinspun. Lifting the winding number: Precise discontinuities in neural fields for physics simulation. In *ACM SIGGRAPH 2025 Conference Papers*, 2025b. doi: 10.1145/3721238.3730597.

Honglin Chen, Rundi Wu, Eitan Grinspun, Changxi Zheng, and Peter Yichen Chen. Implicit neural spatial representations for time-dependent PDEs. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 5162–5177, 2023a.

Peter Yichen Chen, Maurizio M. Chiaramonte, Eitan Grinspun, and Kevin Carlberg. Model reduction for the material point method via an implicit neural representation of the deformation map. *Journal of Computational Physics*, 478:111908, 2023b. doi: 10.1016/j.jcp.2023.111908.

Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5939–5948, 2019.

Yitong Deng, Hong-Xing Yu, Diyang Zhang, Jiajun Wu, and Bo Zhu. Fluid simulation on neural flow maps. *ACM Transactions on Graphics*, 42(6), 2023.

Michael Firman, Neill D. F. Campbell, Lourdes Agapito, and Gabriel J. Brostow. Diversenet: When one right answer is not enough. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5598–5607, 2018. doi: 10.1109/CVPR.2018.00587.

Abner Guzmán-Rivera, Dhruv Batra, and Pushmeet Kohli. Multiple choice learning: Learning to produce multiple structured outputs. In *Advances in Neural Information Processing Systems*, volume 25, pp. 1799–1807, 2012.

Jufeng Han, Shu Wei, Min Wu, Lina Yu, Weijun Li, Linjun Sun, Hong Qin, and Yan Pang. Implicit neural representation with multi-scale sine activation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 21567–21575, 2026. doi: 10.1609/aaai.v40i26.39305.

Wenrui Hao, Xinliang Liu, and Yahong Yang. Newton informed neural operator for solving nonlinear partial differential equations. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 120832–120860. Curran Associates, Inc., 2024. doi: 10.52202/079017-3839. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/dae8afc6b990aa0b3b5efaa096fbd7fa-Paper-Conference.pdf.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Benjamin Holzschuh, Georg Kohl, Florian Redinger, and Nils Thuerey. P3d: Highly scalable 3d neural surrogates for physics simulations with global context. In *International Conference on Learning Representations*, 2026.

Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. Tetrahedral meshing in the wild. *ACM Transactions on Graphics*, 37(4):60:1–60:14, 2018. doi: 10.1145/3197517.3201353.

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

Pranav Jain, Ziyin Qu, Peter Yichen Chen, and Oded Stein. Neural monte carlo fluid simulation. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. doi: 10.1145/3641519.3657438.

Caleb Jore and Jialin Liu. Bifurcation models: Learning set-valued solution maps with weight-tied dynamics. *arXiv preprint arXiv:2605.07277*, 2026.

Navami Kairanda, Marc Habermann, Christian Theobalt, and Vladislav Golyanik. Neuralclothsim: Neural deformation fields meet the thin shell theory. In *Advances in Neural Information Processing Systems*, volume 37, pp. 110533–110572, 2024.

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.

Frantzeska Lavda, Magda Gregorová, and Alexandros Kalousis. Improving VAE generations of multimodal data through data-dependent conditional priors. In *24th European Conference on Artificial Intelligence*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pp. 1254–1261, 2020. doi: 10.3233/FAIA200226.

Stefan Lee, Senthil Purushwalkam, Michael Cogswell, Viresh Ranjan, David J. Crandall, and Dhruv Batra. Stochastic multiple choice learning for training diverse deep ensembles. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Changqing Li and Christopher D. Rahn. Design of continuous backbone, cable-driven robots. *Journal of Mechanical Design*, 124(2):265–271, 2002. doi: 10.1115/1.1447546.

Yifei Li, Yuchen Sun, Pingchuan Ma, Eftychios Sifakis, Tao Du, Bo Zhu, and Wojciech Matusik. NeuralFluid: Neural fluidic system design and control with differentiable simulation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 84944–84967, 2024a. doi: 10.52202/079017-2697.

Yifei Li, Haixu Wu, Zeyi Xu, Tuur Stuyck, and Wojciech Matusik. Neural modular physics for elastic simulation. In *Proceedings of the International Conference on Machine Learning*, 2026.

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

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

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, 2024b. doi: 10.1145/3648506.

Levi E. Lingsch, Mike Yan Michelis, Emmanuel De Bezenac, Sirani M. Perera, Robert K. Katzschmann, and Siddhartha Mishra. Beyond regular grids: Fourier-based neural operators on arbitrary domains. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30610–30629, 2024.

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E. Turner, and Johannes Brandstetter. PDE-refiner: Achieving accurate long rollouts with neural pde solvers. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Mengfei Liu, Yue Chang, Zhecheng Wang, Peter Yichen Chen, and Eitan Grinspun. Precise gradient discontinuities in neural fields for subspace physics. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pp. 1–11, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 979-8-4007-2137-3. doi: 10.1145/3757377.3763810.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

Vismay Modi, Nicholas Sharp, Or Perel, Shinjiro Sueda, and David I. W. Levin. Simplicits: Mesh-free, geometry-agnostic, elastic simulation. *ACM Transactions on Graphics*, 43(4), 2024. doi: 10.1145/3658184.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 165–174, 2019. doi: 10.1109/CVPR.2019.00025.

Quentin Peyron and Jessica Burgner-Kahrs. Stability analysis of tendon driven continuum robots and application to active softening. *IEEE Transactions on Robotics*, 40:85–100, 2024. doi: 10.1109/TRO.2023.3324571.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

Christian Rupprecht, Iro Laina, Robert DiPietro, Maximilian Baust, Federico Tombari, Nassir Navab, and Gregory D. Hager. Learning in an uncertain world: Representing ambiguity through multiple hypotheses. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3591–3600, 2017.

Louis Serrano, Lise Le Boudec, Armand Kassaï Koupaï, Thomas X. Wang, Yuan Yin, Jean-Noël Vittaut, and Patrick Gallinari. Operator learning with neural fields: Tackling pdes on general geometries. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Aliaksandra Shysheya, Cristiana Diaconu, Federico Bergamin, Paris Perdikaris, José Miguel Hernández-Lobato, Richard E. Turner, and Emile Mathieu. On conditional diffusion models for pde simulations. In *Advances in Neural Information Processing Systems*, volume 37, pp. 23246–23300, 2024.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Breannan D. Smith, Fernando de Goes, and Theodore Kim. Stable neo-hookean flesh simulation. *ACM Transactions on Graphics*, 37(2):1–15, 2018. doi: 10.1145/3180491.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, 2015.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, volume 28, pp. 3483–3491, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An extensive benchmark for scientific machine learning. In *36th Conference on Neural Information Processing Systems (NeurIPS 2022) Track on Datasets and Benchmarks*, 2022.

Yuanyuan Tao, Ivan Puhachov, Derek Nowrouzezahrai, and Paul G. Kry. Neural implicit reduced fluid simulation. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 121:1–121:11, 2024. doi: 10.1145/3680528.3687628.

J. M. T. Thompson and G. W. Hunt. *A General Theory of Elastic Stability*. John Wiley & Sons, 1973.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *International Conference on Learning Representations*, 2023.

Eva van Tegelen, George van Voorn, Ioannis N. Athanasiadis, and Peter van Heijster. Neural ordinary differential equations for learning and extrapolating system dynamics across bifurcations. *Chaos*, 35(10):101103, 2025. doi: 10.1063/5.0288264.

Hrishikesh Viswanath, Yue Chang, Aleksey Panas, Julius Berner, Peter Yichen Chen, and Aniket Bera. Learning lagrangian interaction dynamics with sampling-based model order reduction. *Transactions on Machine Learning Research*, 2026.

Caoliwen Wang, Minghao Guo, Siyuan Chen, Heng Zhang, Mengdi Wang, Xingyu Ni, Hanson Sun, Kunyi Wang, Zherong Pan, Kui Wu, Lingjie Liu, Yin Yang, Chenfanfu Jiang, Taku Komura, Wojciech Matusik, and Peter Yichen Chen. WorldParticle: Unified world simulation of lagrangian particle dynamics via transformer. In *SIGGRAPH Asia 2026 Conference Papers*, 2026.

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, 2021. doi: 10.1137/20M1318043.

Lingchen Yang, Byungsoo Kim, Gaspard Zoss, Baran Gözcü, Markus Gross, and Barbara Solenthaler. Implicit neural representation for physics-driven actuated soft bodies. *ACM Transactions on Graphics*, 41(4):1–10, 2022. doi: 10.1145/3528223.3530156.

Haoyang Zheng, Yao Huang, Ziyang Huang, Wenrui Hao, and Guang Lin. HomPINNs: Homotopy physics-informed neural networks for solving the inverse problems of nonlinear differential equations with multiple solutions. *Journal of Computational Physics*, 500:112751, 2024. doi: 10.1016/j.jcp.2023.112751.

Guancheng Zhou, Zelin Zeng, Yisi Luo, Qi Xie, and Deyu Meng. Tucker-fno: Tensor tucker-fourier neural operator and its universal approximation theory. In *International Conference on Learning Representations*, 2026.

A Appendix

A.1 Data Generation

A.1.1 Reaction–diffusion dataset.

We generate the dataset from the periodic one-dimensional reaction–diffusion equation

$$\partial_t u(x,t) = \varepsilon\, \partial_{xx} u(x,t) + a(x)u(x,t) - u(x,t)^3, \qquad x \in [0,1), \tag{17}$$

with $\varepsilon = 2 \times 10^{-4}$. The spatially varying coefficient field is constructed as

$$a(x) = \sum_{i=1}^{2} A_i \exp\bigl[-w_i(x-c_i)^2\bigr] + a_0, \tag{18}$$

with $a_0 = -0.25$. For every coefficient field, the two centers c_i are sampled uniformly from $[0.08, 0.92]$ subject to a minimum separation of 0.12. The amplitudes are independently sampled from $A_i \sim \mathcal{U}(1.2, 1.8)$ and the Gaussian widths from $w_i \sim \mathcal{U}(700, 1400)$. We sample 1024 distinct coefficient fields for training and 128 each for validation and testing. For every field $a(x)$, 32 trajectories are generated from independent hidden perturbations

$$u_0^{\text{hidden}}(x) = 0.05\,\xi(x), \qquad \xi(x) \sim \mathcal{N}(0,1). \tag{19}$$

The resulting dataset therefore contains 32,768 training trajectories, 4,096 validation trajectories, and 4,096 test trajectories. Importantly, the random perturbation is not provided to the learning model: the observable initial-state channel is set to zero, so all 32 trajectories associated with a fixed $a(x)$ have identical observable conditioning but may produce different final solutions.

A.1.2 Cloth Dataset.

For the cloth dataset, we use nonlinear finite-element simulations of deformable solids with the same physical model and numerical solver. Let $\mathbf{x}(\mathbf{X}, t)$ denote the deformed position of a material point $\mathbf{X}$ and $\mathbf{F} = \partial \mathbf{x}/\partial \mathbf{X}$ its deformation gradient. The dynamics are governed by

$$\rho\, \ddot{\mathbf{x}} = \nabla_{\mathbf{X}} \cdot \mathbf{P} + \rho\, \mathbf{g}, \qquad \mathbf{P} = \frac{\partial \Psi}{\partial \mathbf{F}}, \tag{20}$$

where ρ is the material density, $\mathbf{g}$ denotes external acceleration such as gravity, and $\mathbf{P}$ is the first Piola–Kirchhoff stress. We use the same FEM solver as LiCROM (Chang et al., 2023) with the stable Neo-Hookean strain-energy density (Smith et al., 2018)

$$\Psi(\mathbf{F}) = \frac{\mu}{2}\left(\operatorname{tr}(\mathbf{F}^{\mathsf{T}}\mathbf{F}) - 3\right) + \frac{\lambda}{2}\left(J - 1 - \frac{\mu}{\lambda}\right)^2, \qquad J = \det(\mathbf{F}), \tag{21}$$

where μ and λ are the Lamé parameters. The corresponding elastic potential is

$$E_{\text{elastic}} = \int_{\Omega_0} \Psi(\mathbf{F})\, dV. \tag{22}$$

Randomized discretizations. To expose different stable deformation modes without modifying the underlying continuous problem, we generate multiple volumetric discretizations of the same physical domain, as illustrated in Figure 8. We recover the polygonal cross section of each thin extruded geometry and construct randomized surface triangulations by resampling points along boundary and vertical edges and adding random interior samples on the planar faces. All sampled vertices remain exactly on the original piecewise-planar surface, so only the numerical discretization changes while the continuous geometry is preserved. Each surface mesh is then tetrahedralized using TetWild (Hu et al., 2018) at a comparable target resolution. These discretization differences act as small unobserved perturbations that can steer the nonlinear FEM system toward different attraction basins.

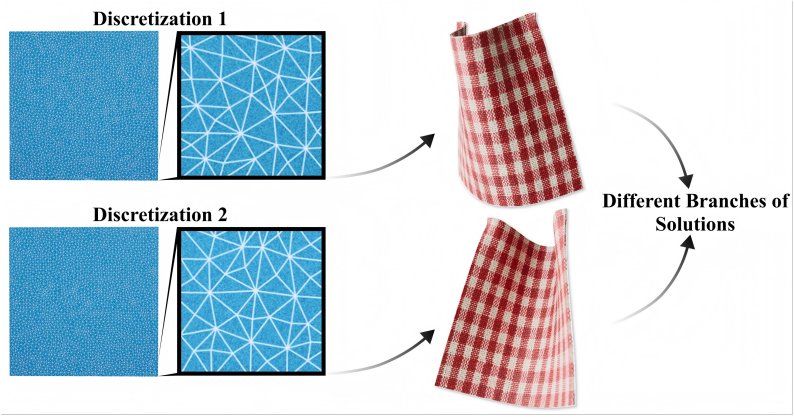

Figure 8: **Randomized discretizations reveal distinct stable outcomes.** Different tetrahedral discretizations of the same cloth geometry are evolved under identical loading and material parameters. Although the continuous problem is unchanged, small discretization differences steer the simulation toward different stable solution branches.

Cloth. We construct thin volumetric cloth domains. For all cloth simulations, we integrate the FEM system for 10,000 time steps with $\Delta t = 1/200$ and record the deformation every 125 steps.

We learn a mapping from material stiffness to the final deformed configuration of the cloth. Two regions on opposite sides of the sheet are constrained and gradually displaced toward the center, creating an in-plane compressive load under gravity. This setup produces two dominant buckling modes, with small discretization differences determining which mode emerges. We vary the material stiffness by setting $\lambda = \mu$ and sweeping their shared value from 1×10^6 to 25×10^6 in increments of 0.5×10^6. For each stiffness value, we simulate the same continuous configuration on four independently generated volumetric meshes, producing multiple possible final states under the same observable condition.

A.1.3 Tendon-driven continuum robot

We reproduce the one-disk planar tendon-driven continuum robot experiment of Peyron & Burgner-Kahrs (2024, Figure 3), illustrated in Figure 9. The robot consists of a flexible backbone of fixed length $L = 0.20\,\mathrm{m}$, actuated by a single tendon tension T_1 ($T_2 = 0$). Following their model, we represent the backbone as a planar Kirchhoff rod and solve its nonlinear static boundary-value problem by single shooting. Equilibrium branches are traced by continuation in T_1; multi-start initialization is used to discover the disconnected branch that cannot be reached by continuing from the unloaded configuration.

We independently classify each equilibrium using the elastic-stability criterion of Peyron & Burgner-Kahrs (2024), retaining only stable configurations for learning. Our reproduction recovers the reported bifurcation structure, including the limit point at approximately $29.7\,\mathrm{N}$. Consequently, the same observable tendon tension can correspond to multiple stable robot shapes.

Importantly, this benchmark is based on a physically evaluated robot model rather than an unconstrained synthetic system. The original study compares its model with measurements from a physical tendon-driven continuum robot and experimentally demonstrates the predicted stability-dependent behavior (Peyron & Burgner-Kahrs, 2024); related experimental validation of continuous-backbone cable-driven robots was also reported by

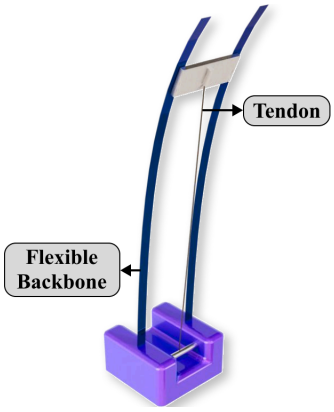

Figure 9: **Tendon-driven continuum robot benchmark.** One-disk planar TDCR considered by Peyron & Burgner-Kahrs (2024). The same tendon tension can admit multiple equilibrium configurations across a fold bifurcation.

Table 4: **Candidate-model configurations used for multi-solution learning.** "Branch dim." denotes the number of learned candidate-feature channels, except for reaction–diffusion diffusion, where the feature is concatenated along the spatial axis.

Dataset	Backbone	K	Branch Dim.	Backbone Configuration
Reaction–Diffusion	FNO	4	8	16 modes, 128 hidden channels
Reaction–Diffusion	Neural Field	4	8	4 hidden layers, 256 hidden features
Reaction–Diffusion	Cond. Diffusion	6	1	U-Net $[62, 128, 256, 1024]$
Cloth	FNO	4	8	16×16 modes, 64 hidden channels
Cloth	Neural Field	4	8	4 hidden layers, 256 hidden features
Cloth	Cond. Diffusion	4	8	U-Net $[32, 64, 128]$
Robot	FNO	4	8	12 modes, 64 hidden channels
Robot	Neural Field	4	8	4 hidden layers, 256 hidden features
Robot	Cond. Diffusion	4	8	U-Net $[32, 64, 128]$

Li & Rahn (2002). We use their validated numerical formulation to generate the learning data, rather than collecting new hardware measurements.

A.2 Multi-Solution Candidate Learning: Implementation Details

Candidate construction. For each backbone, we train a single conditional model with K learned solution features z_k. Given condition a, candidate k is

$$\hat{u}_k(a) = F_\theta(a, z_k), \qquad k = 1, \dots, K. \tag{23}$$

Training uses hard best-match assignment. For each observed pair (a_i, u_i), only the candidate with the smallest reconstruction loss receives the reconstruction gradient,

$$k_i^\star = \arg\min_k \ell\left(\hat{u}_k(a_i), u_i\right). \tag{24}$$

This allows candidate hypotheses to specialize without requiring branch labels.

Injecting solution features. The solution feature is injected differently depending on the backbone. For FNOs, we use a learned smooth Fourier feature field for each candidate and concatenate it with the physical input as additional channels. The feature uses four Fourier frequencies per spatial axis, producing a spatially varying conditioning signal.

For coordinate-based neural fields, each candidate is represented by a learned vector z_k, which is concatenated with the spatial coordinate and encoded physical condition at every query point.

For the cloth and robot diffusion models, we use the same smooth Fourier feature construction as for FNO and concatenate the resulting candidate field with the input channels before the score U-Net. The reaction–diffusion diffusion model uses a one-dimensional variant: each candidate contains one learned bias together with four Fourier sine/cosine amplitude pairs. The resulting feature is broadcast to length 256 and concatenated along the spatial axis rather than the channel axis.

Reconstruction loss. The reconstruction discrepancy ℓ is architecture dependent. For FNO, we use an H^1 loss that includes both solution and finite-difference derivative errors. Neural fields use mean-squared error. Diffusion models use the standard denoising objective,

$$\mathcal{L}_{\text{diff}} = \|\epsilon_\theta(u_t, t, a, z_k) - \epsilon\|_2^2, \tag{25}$$

where ϵ is the injected Gaussian noise.

Candidate separation. Hard best-match training alone does not prevent multiple candidates from collapsing to the same solution. We therefore use a hinge-style separation penalty,

$$\mathcal{L}_{\text{sep}} = \frac{2}{K(K-1)} \sum_{k<k'} \left[m - \|\hat{u}_k(a) - \hat{u}_{k'}(a)\|_2\right]_+^\gamma, \tag{26}$$

Table 5: **Separation-loss parameters used for multi-solution candidate learning.** The separation scale is adjusted for each dataset and model family because prediction distances vary substantially across architectures and tasks.

Dataset	Backbone	λ_{sep}	Margin m
Reaction–Diffusion	FNO / Neural Field	20.0	1.0
Reaction–Diffusion	Cond. Diffusion	8.0	2.0
Cloth	FNO / Neural Field	20.0	1.0
Cloth	Cond. Diffusion	5.0	5.0
Robot	FNO / Neural Field	20.0	1.0
Robot	Cond. Diffusion	0.2	1.0

Table 6: **Optimization settings for multi-solution candidate learning.**

Backbone	Optimizer	Learning Rate	Scheduler	Epochs
FNO	AdamW, wd $= 10^{-4}$	8×10^{-3}	Cosine annealing	250
Neural Field	Adam	10^{-4}	MultiStepLR $\times 0.1$ at $1/3$, $2/3$	250
Cond. Diffusion (Reaction–Diffusion)	AdamW, wd $= 10^{-12}$	10^{-3}	Cosine annealing	5000
Cond. Diffusion (Cloth / Robot)	AdamW, wd $= 10^{-12}$	10^{-3}	Cosine annealing	2000

where m is the separation margin and γ denotes the fixed hinge power. The total candidate-learning objective is

$$\mathcal{L}_{\text{cand}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{sep}} \mathcal{L}_{\text{sep}}. \tag{27}$$

For deterministic FNO and neural-field models, separation is measured directly between candidate outputs. For diffusion models, separation is applied to partially reconstructed clean samples rather than directly in noise-prediction space. All candidate branches use the same starting noise, and separation is evaluated along the same partial-denoising trajectory. This prevents independent noise realizations from being interpreted as genuine branch separation.

Optimization. The optimization settings are summarized in Table 6. FNO models use AdamW and cosine annealing. Neural fields use Adam with a piecewise learning-rate decay and gradient clipping. Diffusion models use AdamW and cosine annealing, with longer training schedules.

For neural fields, we additionally clip the gradient norm to 1.0 to prevent occasional optimization spikes from collapsing the SIREN to a nearly constant solution.

Backbone implementations. Our FNO models are implemented using the `neuraloperator` library (Kossaifi et al., 2025). The coordinate-based neural-field backbone follows SIREN (Sitzmann et al., 2020; Kairanda et al., 2024), using sinusoidal activations to represent the solution continuously over spatial coordinates. For both backbones, we retain the standard underlying architecture and augment the input with the learned solution features described above.

Our diffusion models build on the PDEDiff `AmortizedScoreUNet` of Shysheya et al. (2024), together with a variance-preserving diffusion SDE, and are trained to predict the injected noise. We maintain an exponential moving average of the score-network parameters with decay 0.9999 and use the EMA parameters for evaluation and sampling. For cloth and robot, the physical condition is represented as a broadcast spatial field, whereas reaction–diffusion follows the masked-frame conditioning scheme of the underlying PDEDiff implementation.

Training procedure. Algorithm 1 summarizes the learning of solution hypotheses. We jointly optimize the shared physics model and the K learned solution features using hard best-match assignment, so that each observation updates only its closest candidate. A separation term discourages different hypotheses from collapsing to the same solution. After this optimization, both the physics model and solution features are frozen before training the condition-dependent activation model.

Algorithm 1 Learning Solution Hypotheses

Require: Training data $\mathcal{D} = \{(a_i, u_i)\}_{i=1}^N$; candidate model F_θ; K learnable solution features $\{z_k\}_{k=1}^K$; reconstruction loss ℓ; separation weight λ_{sep}; margin m
Ensure: Frozen candidate model F_θ and solution features $\{z_k\}$
Initialize θ and $\{z_k\}_{k=1}^K$
for each training minibatch $\mathcal{B} \subset \mathcal{D}$ **do**
  for each $(a_i, u_i) \in \mathcal{B}$ **do**
    for $k = 1, \dots, K$ **do**
      Compute candidate prediction $\hat{u}_{ik} \leftarrow F_\theta(a_i, z_k)$
      Compute architecture-specific discrepancy $\ell_{ik} \leftarrow \ell(\hat{u}_{ik}, u_i)$
    end for
    Assign observation to its best-matching hypothesis:

$$k_i^\star \leftarrow \arg\min_k \ell_{ik}$$

  end for
  Compute best-match reconstruction loss:

$$\mathcal{L}_{\text{fit}} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \ell_{ik_i^\star}$$

  Compute pairwise candidate-separation loss:

$$\mathcal{L}_{\text{sep}} \leftarrow \frac{2}{|\mathcal{B}|K(K-1)} \sum_{i \in \mathcal{B}} \sum_{k<k'} [m - \|\hat{u}_{ik} - \hat{u}_{ik'}\|_2]_+^q$$

  $\mathcal{L}_{\text{hyp}} \leftarrow \mathcal{L}_{\text{fit}} + \lambda_{\text{sep}} \mathcal{L}_{\text{sep}}$
  Update θ and $\{z_k\}$ using $\nabla \mathcal{L}_{\text{hyp}}$
end for
Freeze θ and $\{z_k\}_{k=1}^K$

Freezing. After candidate learning, both the backbone parameters θ and the learned solution features $\{z_k\}_{k=1}^K$ are frozen. The subsequent activation model therefore operates on a fixed collection of candidate solutions and cannot modify the multi-solution predictor.

A.3 Condition-Dependent Activation: Implementation Details

Activation network. The candidate solution model is frozen while training the activation model. The activation network receives the same condition field used by the candidate model, rather than a privileged scalar representation. For reaction–diffusion, the input is the forcing field $a(x) \in \mathbb{R}^{128}$. For cloth, the material stiffness is broadcast to a constant spatial field. For the tendon-driven robot, the condition $(L, T_1/50)$ is broadcast as a two-channel field over the 64 spatial nodes.

We use a small convolutional network to predict the K activation logits. For one-dimensional inputs, the architecture is

$$\begin{aligned}&\text{Conv1D}(C_{\text{in}}, 16, 5) \to \text{ReLU} \to \text{Conv1D}(16, 32, 5) \to \text{ReLU} \\ &\quad \to \text{AdaptiveAvgPool1D}(1) \to \text{Linear}(32, K).\end{aligned}$$

with padding 2 for both convolutional layers. Cloth uses the corresponding two-dimensional convolutions and adaptive average pooling. The resulting classifier contains only approximately 10^3–10^4 parameters.

Coverage objective. For training example (a_i, u_i), let $\hat{u}_k(a_i)$ denote the frozen prediction of candidate k, and define

$$d_{ik} = \|\hat{u}_k(a_i) - u_i\|_2^2 . \tag{28}$$

The activation probabilities are

$$\mathbf{p}(a_i) = \sigma(G_\phi(a_i)) = [p_1(a_i), \dots, p_K(a_i)] . \tag{29}$$

We use the soft coverage loss

$$\mathcal{L}_{\text{cov}} = -\frac{1}{N}\sum_{i=1}^{N}\log\left[\sum_{k=1}^{K} p_k(a_i)\exp\left(-\frac{d_{ik}}{T}\right)\right], \tag{30}$$

which encourages at least one highly activated candidate to explain each observation.

The coverage temperature is estimated from the frozen candidate model:

$$T = 3\ \text{median}_i\left(\min_k d_{ik}\right). \tag{31}$$

Target solution count. A uniform sparsity penalty can suppress valid simultaneous branches when a condition genuinely admits multiple solutions. We therefore estimate a condition-dependent target count M_i and penalize only activation beyond this budget. We emphasize that this automatically estimated training target M_i is distinct from the manually annotated M_{true} used only for branch-count evaluation in Section 5.3.

The target count is computed entirely from the raw training observations, without using ground-truth branch identities or predictions from the candidate model. We first group samples with the same physical condition. Within each group, the observed outputs are clustered using connected components under a dataset-specific pairwise ℓ_2 distance threshold. To reduce under-counting when an exact condition contains only a small number of repeated observations, we pool each condition group with its five nearest neighboring condition groups, measured in condition-field distance, and repeat the clustering. The resulting number of clusters is used as M_i for all samples associated with that condition.

The corresponding excess-only sparsity loss is

$$\mathcal{L}_{\text{sparse}} = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{k=1}^{K} p_k(a_i) - M_i\right]_+, \tag{32}$$

where

$$[x]_+ = \max(x, 0). \tag{33}$$

Thus, up to M_i active hypotheses are unpenalized, while additional activation is discouraged.

Duplicate suppression. The count constraint alone does not prevent several active hypotheses from representing nearly identical solutions. We therefore use

$$\mathcal{L}_{\text{div}} = \frac{2}{NK(K-1)}\sum_{i=1}^{N}\sum_{k<k'} p_k(a_i)p_{k'}(a_i)\exp\left(-\frac{\|\hat{u}_k(a_i) - \hat{u}_{k'}(a_i)\|_2^2}{\sigma^2}\right), \tag{34}$$

which penalizes pairs of candidates that are simultaneously active and close in solution space. The kernel width is estimated from the frozen candidate predictions as

$$\sigma = \text{median}\left\{\|\hat{u}_k(a_i) - \hat{u}_{k'}(a_i)\|_2 : k < k'\right\}. \tag{35}$$

The complete activation objective is

$$\mathcal{L}_{\text{act}} = \mathcal{L}_{\text{cov}} + \lambda_{\text{sparse}}\mathcal{L}_{\text{sparse}} + \lambda_{\text{div}}\mathcal{L}_{\text{div}}. \tag{36}$$

Both weights are fixed throughout training. We use $\lambda_{\text{sparse}} = 1.0$ and $\lambda_{\text{div}} = 2.5$ for the results in Tables 2 and 3, with an ablation provided in Section A.8. No warm-up schedule is used.

Optimization. Only the activation-network parameters ϕ are optimized. The candidate model and its solution hypotheses remain frozen throughout training. We use AdamW with weight decay 10^{-4} and cosine learning-rate decay over 300 epochs. The learning rate is 3×10^{-3} for FNO and neural-field models and 10^{-3} for diffusion models. Batch sizes are 64 for FNO and neural fields and 128 for diffusion.

Algorithm 2 Condition-Dependent Solution Activation

Require: Training data $\{(a_i, u_i)\}$; frozen candidates $\{\hat{u}_k\}_{k=1}^{K}$; target counts $\{M_i\}$; activation model G_ϕ
Ensure: Trained activation model G_ϕ
for each minibatch $\mathcal{B}$ **do**
  for each $(a_i, u_i) \in \mathcal{B}$ **do**
    $\mathbf{p}_i \leftarrow \sigma(G_\phi(a_i))$
    $d_{ik} \leftarrow d(u_i, \hat{u}_k(a_i)), \quad k = 1, \ldots, K$
  end for
  $\mathcal{L}_{\text{cov}} \leftarrow -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \log \sum_k p_{ik} \exp(-d_{ik}/T)$
  $\mathcal{L}_{\text{sparse}} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \left[\sum_k p_{ik} - M_i \right]_+$
  $\mathcal{L}_{\text{div}} \leftarrow \frac{2}{|\mathcal{B}| K(K-1)} \sum_{i \in \mathcal{B}} \sum_{k<k'} p_{ik} p_{ik'} \exp\left(-\frac{\|\hat{u}_k(a_i) - \hat{u}_{k'}(a_i)\|_2^2}{\sigma^2} \right)$
  $\mathcal{L}_{\text{act}} \leftarrow \mathcal{L}_{\text{cov}} + \lambda_{\text{sparse}} \mathcal{L}_{\text{sparse}} + \lambda_{\text{div}} \mathcal{L}_{\text{div}}$
  Update ϕ using $\nabla_\phi \mathcal{L}_{\text{act}}$
end for
At inference, retain $\{k : p_k(a) \geq 0.5\}$; if empty, retain $\arg\max_k p_k(a)$

Diffusion candidates. Unlike deterministic backbones, a frozen diffusion hypothesis does not produce a unique output. For each condition and hypothesis, we therefore precompute 50 independent denoising samples using DPM-Solver++ Lu et al. (2022) with 16 steps and first-order updates. During activation training, one cached sample is selected randomly for each training instance. This exposes the activation model to the sampling variability of the frozen diffusion model rather than to a single fixed realization.

Activation training. Algorithm 2 summarizes condition-dependent solution activation. With the learned candidate model fixed, we first estimate a target solution multiplicity from repeated observations at the same and nearby conditions. The activation network then predicts independent probabilities for the K frozen hypotheses and is trained using a soft coverage objective, an excess-only count penalty, and a duplicate-suppression term. At inference, thresholding these probabilities converts the fixed candidate budget into a condition-dependent predicted solution set.

Inference. At test time, candidate k is retained when

$$p_k(a) \geq 0.5. \tag{37}$$

If no candidate exceeds the threshold, we retain the highest-probability candidate so that the predicted solution set is never empty. Evaluation uses the resulting active set for branch-count error and symmetric set distance; the corresponding "without activation" baseline retains all K frozen candidates, thereby isolating the contribution of condition-dependent activation from that of multi-solution candidate learning.

A.4 Sampling Efficiency

We compare branch-conditioned and standard conditional diffusion under matched sampling budgets. At round r, our method draws r samples from each of the K active branches, while the baseline draws rK independent samples. We report the minimum MSE over the resulting samples.

As shown in Figure 10, branch conditioning reaches low-error solutions with substantially fewer samples across all three benchmarks. With larger budgets, vanilla diffusion catches up on some datasets, indicating that the main benefit is more direct and sample-efficient access to distinct solution modes rather than increased asymptotic expressiveness.

A.5 Branch-aware inverse inference.

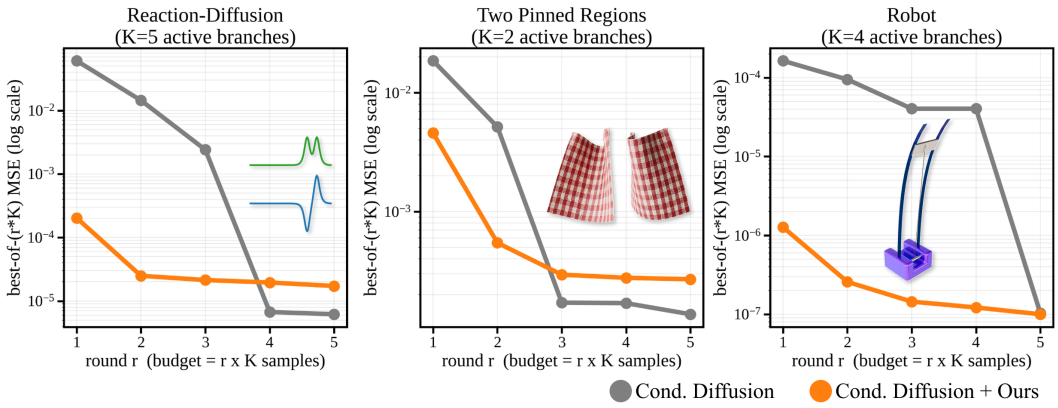

Figure 10: **Sampling efficiency of branch-conditioned vs. vanilla conditional diffusion across all three benchmarks.** At round r, vanilla draws $r \times K$ independent samples and ours draws r samples from each of K active branches, same total budget, minimum MSE reported for both. Our method shows strong sampling efficiency, especially at low sample budgets.

Following Li et al. (2021), who demonstrated the use of an FNO surrogate for a downstream inverse problem, we consider recovering the tendon force T_1 from a partial observation of the robot configuration. For each observation, we search for the force whose predicted tip position best matches the observed one. As shown in Table 7, before the fold the forward map is effectively single-valued and all methods achieve similarly small median errors. After the fold, multiple stable configurations coexist for the same force, making solution admissibility important: without activation, an otherwise inactive hypothesis can provide a spurious match during the inverse search. Condition-dependent activation reduces the post-fold median error from $5.06\,\mathrm{N}$ to $1.58\,\mathrm{N}$, compared with $6.51\,\mathrm{N}$ for the vanilla FNO. This result illustrates that activation is useful not only for recovering the admissible solution set, but also for downstream inference once the physical system becomes multistable.

Table 7: **Tendon-force recovery from partial robot observations.** Median absolute error in T_1 (N, ↓), reported before and after the fold.

Region	Vanilla	Ours w/o Activation	Ours + Activation
Pre-fold	**0.03**	0.10	0.04
Post-fold	6.51	5.06	**1.58**

A.6 Additional Qualitative Results

A.6.1 Deterministic Models

Reaction-Diffusion Dataset Figure 11 shows qualitative predictions for three different input conditions $a(x)$. For both deterministic backbones, the vanilla models produce a single averaged response and fail to represent the multiple admissible outcomes. In contrast, our formulation learns several candidate solutions for the same input, with different hypotheses recovering distinct branches of the solution set. These examples complement the quantitative results reported in the main text.

Cloth Dataset Figure 12 shows qualitative results for the two-pinned-region cloth benchmark, where the input material condition is specified by the Lamé stiffness parameters and the target is the final deformed configuration. The first row corresponds to a lower stiffness setting (3×10^6), while the second uses a higher stiffness setting (25×10^6). The vanilla neural field fails to recover plausible deformations, while the vanilla FNO produces plausible predictions but captures only one of the valid deformation modes. In contrast, our formulation enables both deterministic backbones to represent multiple distinct deformation modes under the same material condition. Quantitative results are reported in the main text.

Tendon-driven Continuum Robot Dataset. The tendon-driven robot provides a useful example of why a fixed collection of solution hypotheses alone is insufficient for representing a condition-dependent solution set. The hypothesis budget K must be large enough to accommodate the maxi-

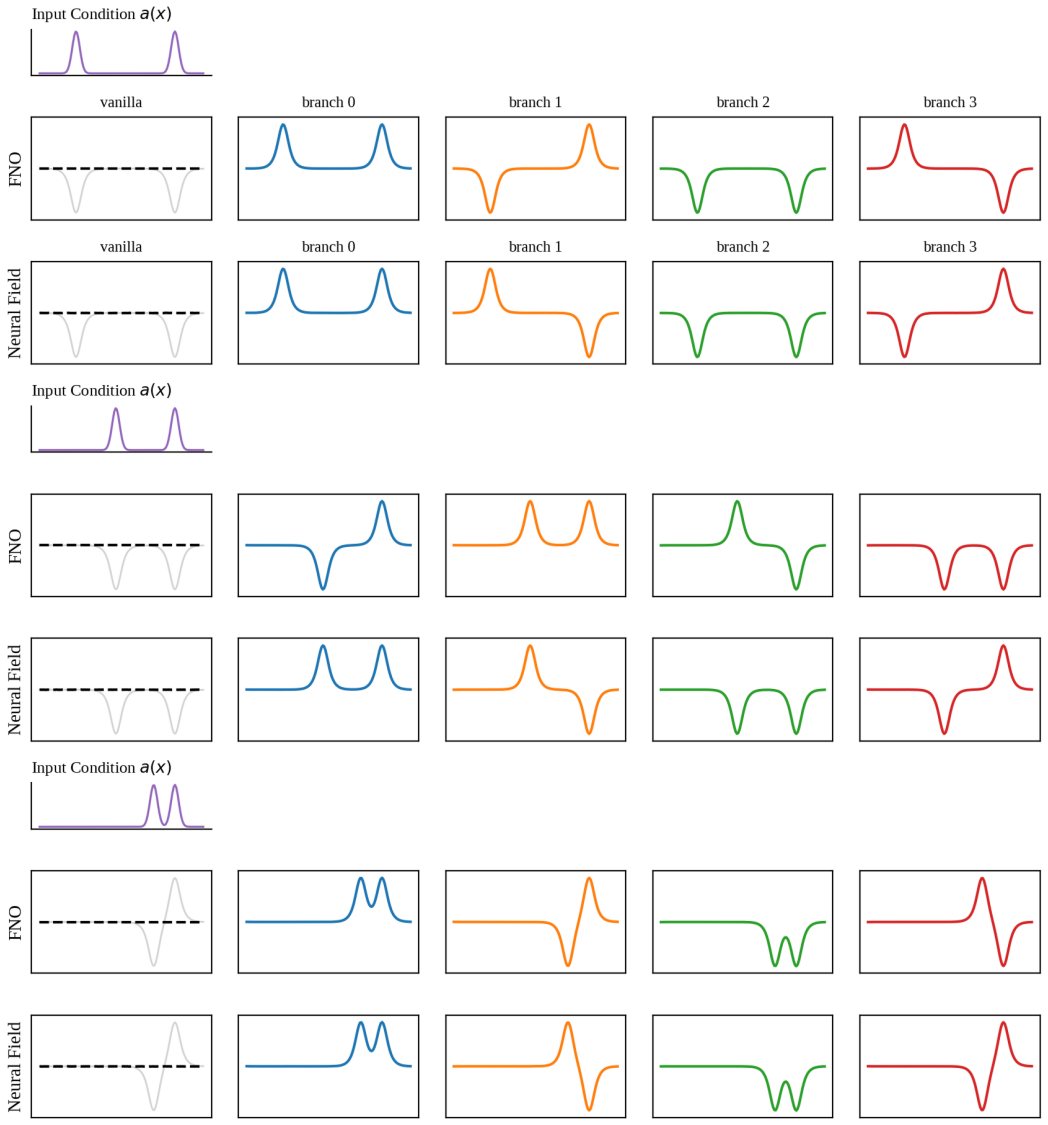

Figure 11: **Qualitative comparison of deterministic models on the reaction–diffusion benchmark.** We visualize predictions for three different input conditions $a(x)$. For both FNO and neural-field backbones, the vanilla deterministic models collapse toward an averaged solution and fail to represent the multiple valid outcomes. In contrast, our formulation produces multiple candidate solutions that capture distinct branches of the solution set for each input condition.

mum solution multiplicity, but at a particular loading condition not all candidates need be admissible. As shown in Figure 13, extra hypotheses may duplicate the same physical branch or produce configurations that do not correspond to valid equilibria, while the true number of admissible solutions changes from one to three across the loading range. Condition-dependent activation is therefore used to select, from the fixed candidate pool, the subset representing the solution set at each condition.

Figure 15 shows the resulting activated solutions for the neural-field and FNO backbones. Both recover plausible equilibrium configurations throughout the loading range, including the transition from one to three admissible solutions. A notable difference is how the learned hypotheses are organized across conditions. For the neural field, hypothesis identities remain largely continuous as T_1 varies, so that the same hypothesis can often be followed along a physical solution branch. The FNO does not exhibit the same consistency: hypothesis indices can switch between physical

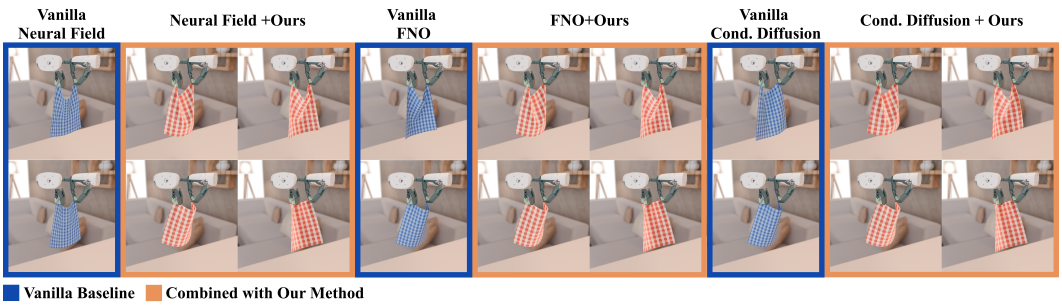

Figure 12: **Qualitative comparison on the two-pinned-region cloth benchmark.** The task maps Lamé stiffness parameters to the final cloth deformation. The first row corresponds to a lower stiffness setting (3×10^6) and the second to a higher one (25×10^6). The vanilla neural field fails to produce plausible deformations, while the vanilla FNO captures only one valid mode and conditional diffusion produces plausible stochastic samples. Our formulation recovers multiple distinct deformation modes under the same material condition.

branches as the loading changes. This does not prevent solution-set recovery because the hypotheses are not assigned predefined global branch identities.

Figure 14 shows the corresponding activation scores. At lower tensions, only one hypothesis is retained, while additional hypotheses become active as multiple equilibria emerge. The activation model also adapts when branch assignments change, as observed more clearly for the FNO and diffusion backbones. Thus, activation does not assume a globally consistent correspondence between hypothesis index and physical branch; it instead identifies which of the currently represented hypotheses are admissible at each condition.

A.6.2 Conditional Diffusion Model

Reaction-Diffusion Dataset Figure 16 provides qualitative diffusion results for the reaction–diffusion benchmark using five independent random seeds. Beyond showing that the model generates valid solution samples, we observe an additional specialization behavior: vanilla conditional diffusion can reach different solution modes across seeds, whereas conditioning on a fixed learned hypothesis tends to concentrate samples around a particular mode with less cross-mode variation. Different hypotheses favor different valid solutions, qualitatively matching the specialization behavior suggested by our theoretical analysis in Section A.9.

Cloth Dataset We provide qualitative comparisons on the cloth benchmark. For the two-pinned-region setting, where opposite sides of the sheet are moved inward to induce buckling, our method produces visually plausible final deformations, as shown in Figure 12.

Beyond demonstrating that the model generates valid solution samples, Figure 17 reveals an additional specialization behavior. Vanilla conditional diffusion may reach different solution modes across random seeds, whereas conditioning on a fixed learned hypothesis tends to concentrate samples around a particular mode with reduced cross-mode variation. Different hypotheses in turn favor different valid solutions, consistent with the specialization behavior predicted by our theoretical analysis.

Tendon-driven Continuum Robot Dataset The conditional diffusion model captures plausible equilibrium configurations and the same one-to-three change in solution multiplicity, shown in Figure 15. However, a fixed hypothesis index does not necessarily correspond to the same physical branch across loading conditions, and branch identities can switch as T_1 varies. This is consistent with the role of the hypothesis as a conditioning variable for a stochastic generative model rather than as an explicit globally tracked physical branch.

A.7 Compatibility with Single-Solution Learning

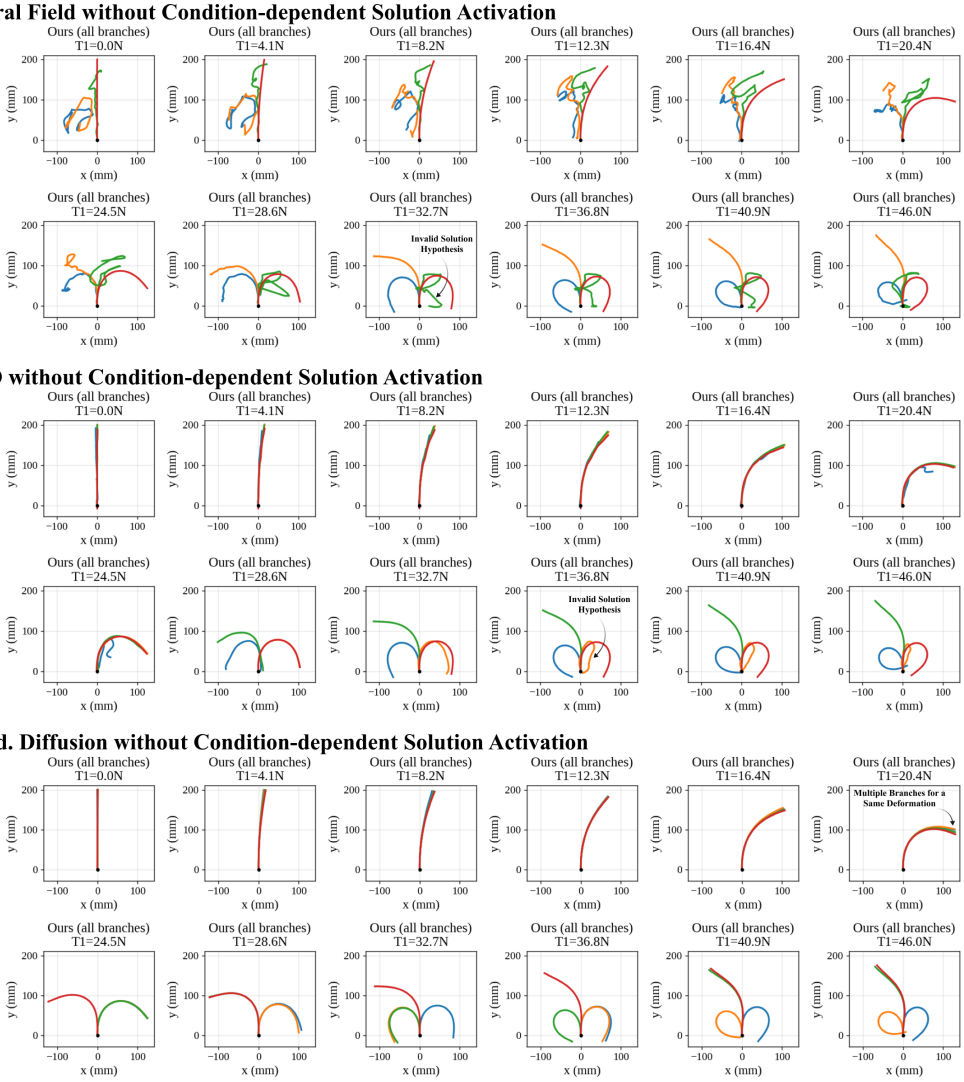

Figure 13: **Why condition-dependent solution activation is needed.** We visualize all learned solution hypotheses for the tendon-driven robot across increasing tendon loads. A fixed hypothesis budget is useful for representing the maximum solution multiplicity, but not every hypothesis is admissible at every condition: some candidates can be redundant, correspond to the same physical branch, or produce invalid intermediate configurations. The number of valid solutions also changes with loading. Condition-dependent activation therefore filters the fixed candidate pool into a smaller admissible solution set for each physical condition.

We additionally evaluate our method on the single-solution Navier–Stokes benchmark from PDEBench (Takamoto et al., 2022). Table 8 compares the vanilla predictor with two evaluations of the learned hypotheses. Best-of-K reports the minimum error over all candidate predictions and therefore measures whether at least one hypothesis remains accurate. Mean Active averages the per-sample MSE over all candidates with $p_k(a) > 0.5$; if no candidate exceeds the threshold, we retain the highest-confidence candidate $\arg\max_k p_k(a)$. This provides a stricter measure of whether the predicted active set as a whole remains compatible with a single-solution problem. Neural fields retain per-

Table 8: **Single-solution Navier–Stokes evaluation. Test-set MSE (↓).**

Method	Vanilla	Best-of-K	Mean Active
FNO	0.0250	0.0256	0.0887
Neural Field	0.0841	**0.0776**	0.0812
Diffusion	0.0910	**0.0232**	0.0262

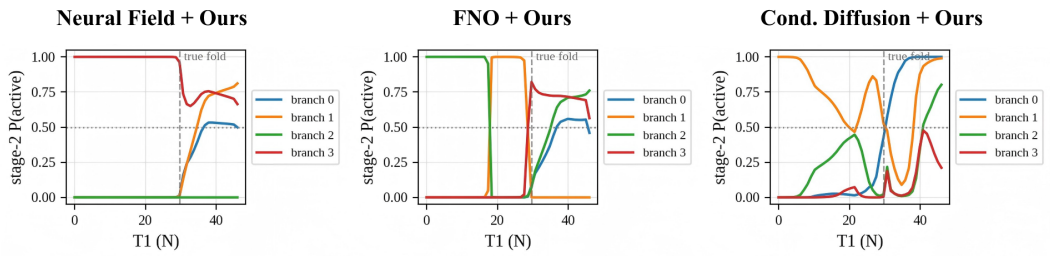

Figure 14: **Condition-dependent activation on the tendon-driven robot.** Predicted activation probabilities for the learned solution hypotheses as the tendon tension T_1 varies. The activation model captures the change in solution multiplicity across the loading range and adapts when the identity of the hypothesis representing a physical branch changes. The dashed vertical line indicates the reference fold location, and the horizontal line marks the activation threshold.

Table 9: **Ablation of the separation regularizer.** We vary the separation-loss weight λ_{sep} and margin m relative to their default values and report best-of-K test MSE (↓). "Off" disables the separation term. The effect of the regularizer is architecture- and dataset-dependent, with no single setting performing best across all cases. A dash denotes experiments not run.

	Regularizer Setting		Reaction–Diffusion			Robot		
Config.	λ_{sep}	m	FNO	Field	Diffusion	FNO	Field	Diffusion
Off	0	–	4.38×10^{-3}	4.80×10^{-6}	4.73×10^{-3}	1.36×10^{-7}	1.51×10^{-7}	1.25×10^{-6}
Margin $\times0.2$	$1\times$	$0.2\times$	2.96×10^{-4}	4.71×10^{-6}	–	3.15×10^{-7}	1.63×10^{-7}	1.77×10^{-6}
Default	$1\times$	$1\times$	4.38×10^{-3}	6.00×10^{-6}	6.57×10^{-5}	1.40×10^{-7}	1.80×10^{-7}	1.60×10^{-6}
Margin $\times5$	$1\times$	$5\times$	1.12×10^{-5}	4.81×10^{-6}	4.31×10^{-1}	1.99×10^{-7}	1.11×10^{-7}	1.47×10^{-6}
Weight $\times0.2$	$0.2\times$	$1\times$	6.43×10^{-2}	4.61×10^{-6}	–	3.38×10^{-7}	1.56×10^{-7}	1.88×10^{-6}
Weight $\times5$	$5\times$	$1\times$	8.25×10^{-6}	4.64×10^{-6}	–	8.17×10^{-7}	1.98×10^{-7}	1.33×10^{-6}

formance comparable to the vanilla model, while diffusion benefits substantially from the multi-hypothesis formulation. For FNO, the activated set degrades prediction quality, highlighting a failure case in which redundant hypotheses are not sufficiently suppressed.

A.8 Ablation Studies

Ablation on the number of hypotheses K. We ablate the hypothesis budget K, which controls the maximum number of candidate solutions represented by the model. If K is smaller than the true solution multiplicity, the full solution set cannot in principle be recovered. Consistent with this, Figure 18 shows a sharp reduction in best-of-K error as K approaches the number of solution branches in both benchmarks. Notably, increasing K already improves accuracy before this point, suggesting that additional hypotheses help the model represent a larger portion of the multimodal solution set.

Increasing K beyond the observed number of branches can sometimes further reduce error, although the effect is model dependent. One possible explanation is that redundant hypotheses allow different predictors to specialize to subsets of the solution space, while a larger hypothesis set can also make optimization more difficult, as seen for the neural-field model in some settings. In practice, these results suggest choosing K comparable to the maximum expected solution multiplicity. When this number is unknown, a smaller K can still provide substantial gains over a single-output model.

Separation regularization. We ablate the weight λ_{sep} and margin m of the separation loss in Table 9.

Their effect depends on both the architecture and dataset. For example, FNO on reaction–diffusion benefits substantially from stronger separation, whereas the neural-field results are relatively insensitive. Diffusion is more sensitive, and an excessively large margin can substantially degrade performance. On the robot benchmark, the different settings remain within a relatively narrow range.

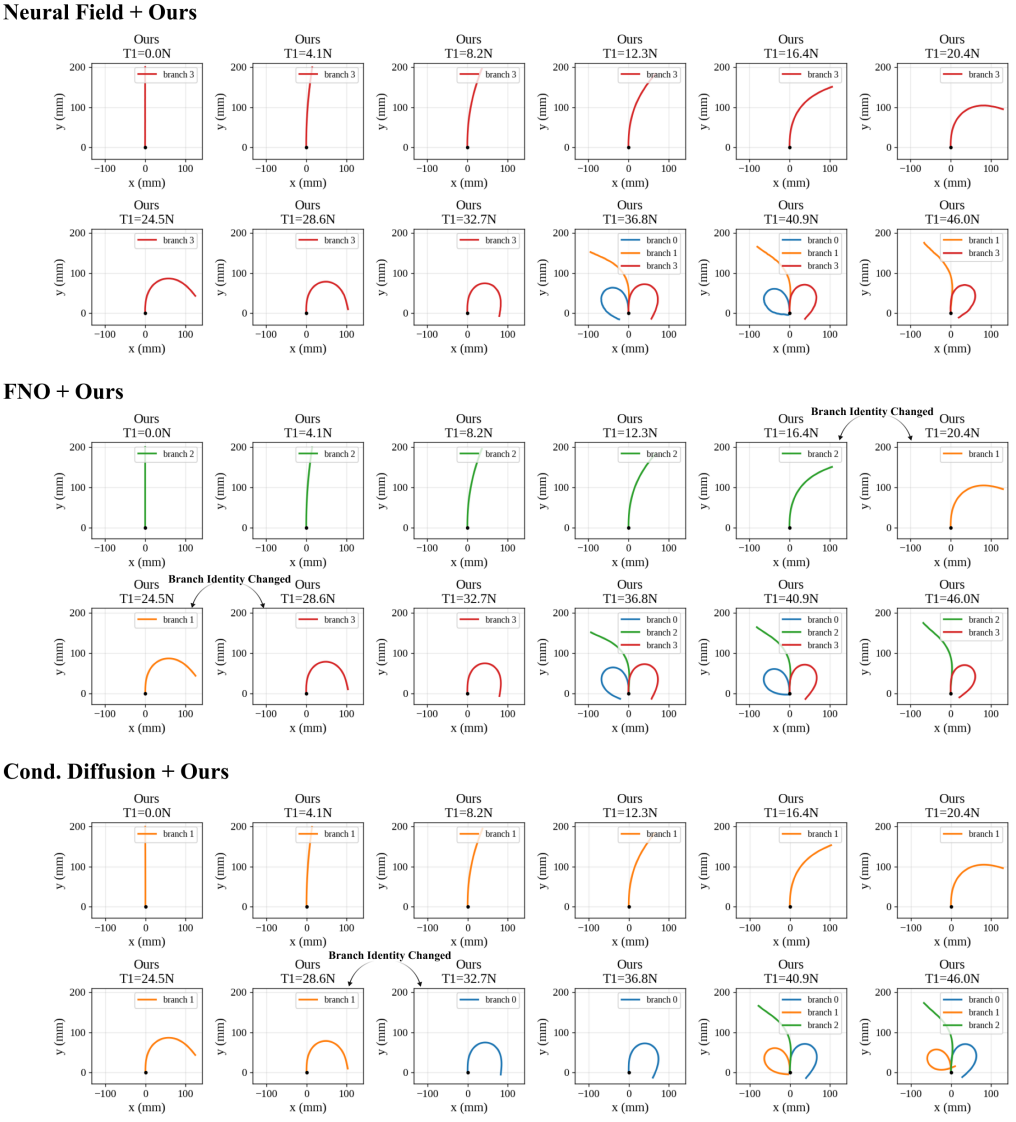

Figure 15: **Predicted equilibrium configurations of the tendon-driven continuum robot.** We show the admissible final configurations under increasing tendon tension T_1 for neural-field, FNO, and conditional-diffusion backbones. All three models recover plausible deformations and the transition from a single solution to three coexisting solutions at larger loads. For the neural field, hypothesis identities remain largely consistent across loading conditions, whereas FNO and conditional diffusion can assign the same physical branch to different hypothesis indices as T_1 varies.

Overall, there is no universal choice of λ_{sep} and m. Most tested settings still outperform the corresponding vanilla baselines, suggesting that the gains are not tied to precise tuning of the separation regularizer. The main exception is the $\times 5$ margin for diffusion on reaction–diffusion, which substantially degrades performance. We therefore treat these quantities as architecture- and dataset-dependent regularization hyperparameters and tune them coarsely in practice.

Ablation on solution-hypothesis architecture F_θ. We ablate how the K solution hypotheses are represented within F_θ. Our default design uses a shared model conditioned on a learned solution feature z_k. We compare against two alternatives from prior multiple-hypothesis work: (i) expanding the output layer to predict K solutions directly, following Rupprecht et al. (2017), and (ii) replacing the learned solution features with fixed categorical features, as in Firman et al. (2018).

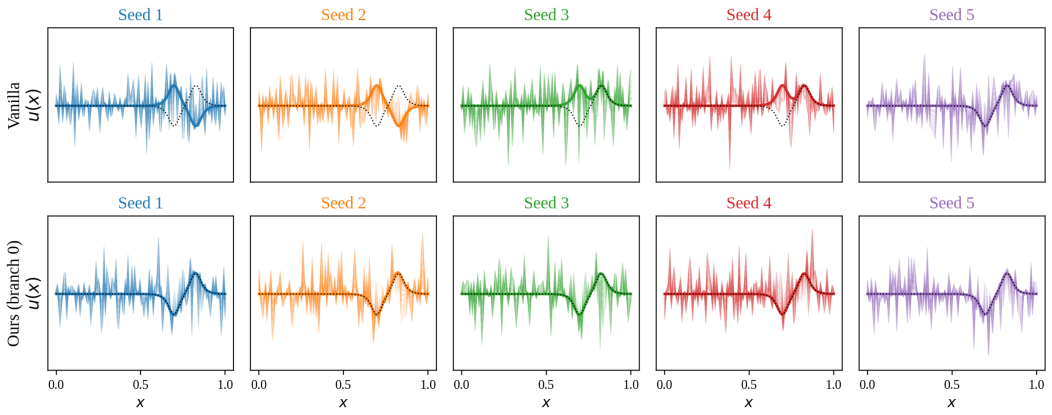

Figure 16: **Branch specialization in diffusion on reaction–diffusion.** We visualize the denoising process from five independent random seeds; faint curves show intermediate denoising states and the solid curve shows the final prediction. Vanilla conditional diffusion can reach different outcomes across seeds. In contrast, conditioning on the same learned solution hypothesis tends to concentrate samples around the same solution mode, with substantially less cross-mode variation. This behavior is consistent with our theoretical analysis, where specialization encourages different hypotheses to focus on complementary members of the solution set.

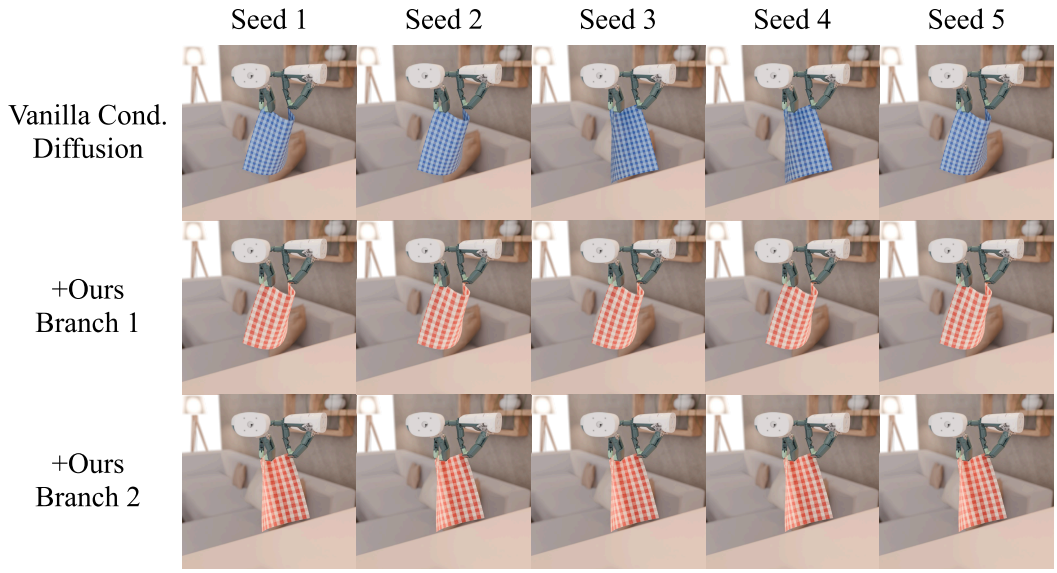

Figure 17: **Branch specialization in diffusion on the two-pinned cloth benchmark.** We show final predictions from five independent diffusion seeds. Vanilla conditional diffusion exhibits greater variation across samples, whereas conditioning on the same learned hypothesis tends to produce the same deformation mode more consistently. Different hypotheses favor different valid modes, illustrating the same specialization behavior observed in reaction–diffusion.

All variants use the same best-match training objective, and we report best-of-K test error.

As shown in Table 10, the learned-feature parameterization gives the lowest error for all three backbones in this experiment. The differences are small for the neural field, while the replicated-output and fixed-feature vari-

Table 10: **Ablation on solution-hypothesis parameterization.**

Method	Ours	$K\times$ Bigger Output Head	Harcoded Feature
FNO	1.4×10^{-7}	5.4×10^{-4}	2.1×10^{-7}
Neural Field	1.0×10^{-7}	1.6×10^{-7}	1.1×10^{-7}
Diffusion	1.6×10^{-6}	1.7×10^{-5}	7.8×10^{-2}

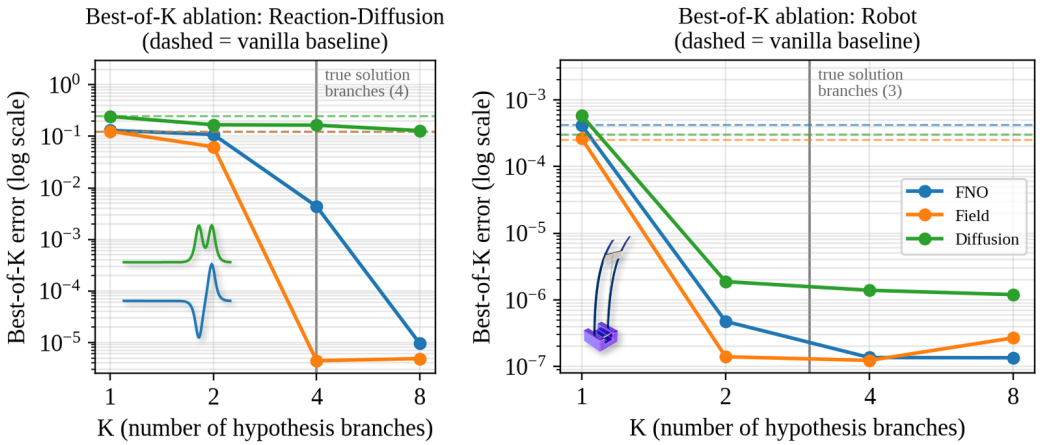

Figure 18: **Ablation on the number of solution hypotheses K.** Increasing K generally reduces best-of-K error as the hypothesis budget approaches the number of true solution branches (vertical gray line). Even when K is smaller than the true multiplicity, additional hypotheses already improve prediction accuracy. Increasing K further can provide additional gains, although the effect depends on the backbone. Dashed lines show the corresponding vanilla baselines.

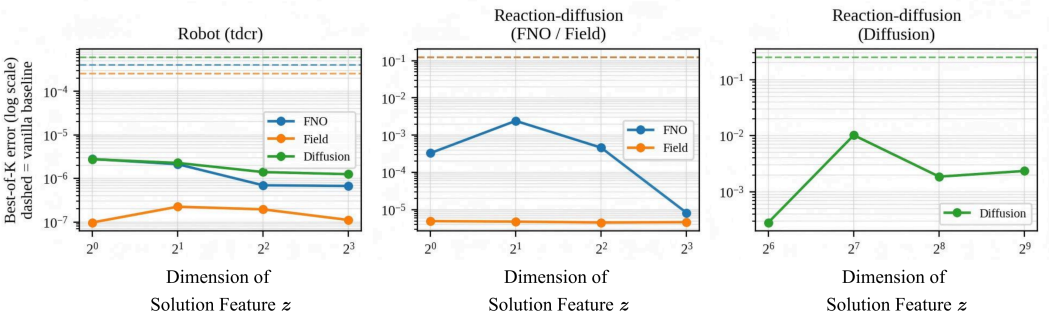

Figure 19: **Ablation on the dimension of the solution feature z.** We vary the dimensionality of the learned solution feature across the robot and reaction–diffusion benchmarks. No consistent monotonic trend is observed, indicating that the preferred feature dimension depends on the architecture and dataset. Nevertheless, all tested dimensions outperform the corresponding vanilla baselines (dashed lines) in these multi-solution settings, suggesting that the method is relatively robust to this hyperparameter.

ants degrade more noticeably for FNO and diffusion. Within the architectures and datasets evaluated here, these results support using learned solution features as our default parameterization.

Ablation on the solution-feature dimension. Figure 19 studies the dimensionality of the learned solution feature z. We do not observe a consistent monotonic trend: increasing the feature dimension can either improve or slightly degrade performance depending on the dataset and backbone. This suggests that the preferred dimension is architecture- and problem-dependent rather than governed by a single universal choice. Importantly, all tested dimensions still outperform the corresponding vanilla baselines in these multi-solution settings, indicating that the method is relatively insensitive to the precise size of z over the tested range.

Sensitivity to Activation Regularization Figure 20 evaluates the sensitivity of the activation model to the sparsity and diversity weights. We train all 3×3 combinations of $\lambda_{\text{sparse}} \in \{1.0, 2.5, 5.0\}$ and $\lambda_{\text{div}} \in \{0.5, 1.25, 2.5\}$ for each of the nine dataset–backbone combinations, resulting in 81 independent trainings. The resulting branch-count accuracy is generally insensitive

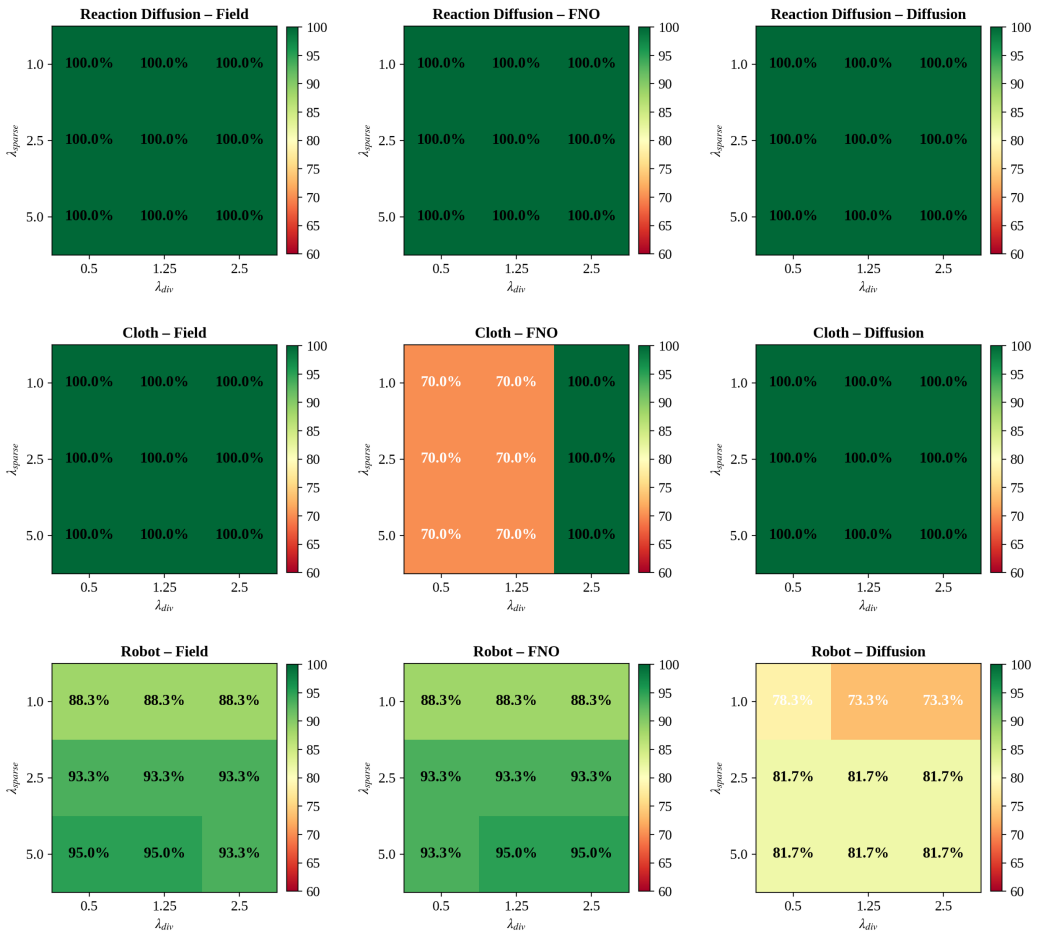

Figure 20: **Sensitivity of condition-dependent activation to sparsity and diversity weights.** We sweep $\lambda_{\text{sparse}} \in \{1.0, 2.5, 5.0\}$ and $\lambda_{\text{div}} \in \{0.5, 1.25, 2.5\}$ for each combination of three datasets and three backbone architectures, for a total of 81 independently trained activation models. Each cell reports test-set exact branch-count accuracy (%). Performance is largely stable across the grid: reaction–diffusion achieves 100% accuracy for all settings, as do the neural-field and diffusion models on cloth. Cloth–FNO benefits from a larger diversity weight, while the robot benchmark shows somewhat greater sensitivity to the sparsity weight. Overall, the activation model performs well over a broad range of hyperparameters rather than relying on a narrowly tuned configuration.

to the precise choice of weights. Reaction–diffusion reaches 100% exact-count accuracy throughout the grid, while the neural-field and diffusion models on cloth are also unchanged across all settings. Cloth–FNO improves from 70% to 100% when the diversity weight is increased to 2.5. The robot benchmark is more challenging, but remains relatively stable: stronger sparsity generally improves the deterministic backbones to approximately 95% accuracy, while the diffusion model varies within a comparatively narrow range of 73.3–81.7%. These results indicate that the activation stage is robust over a broad range of regularization strengths and does not depend on a carefully selected hyperparameter pair.

Robustness of target solution-count estimation. The sparsity term in our activation objective uses a target solution count M_i (see Section A.3) to discourage retaining more active hypotheses than are supported by the observations. Estimating M_i from observations at exactly the same condition, however, can underestimate the true multiplicity when only a subset of valid solutions is observed. We therefore estimate M_i by additionally pooling observations from nearby conditions before clustering, exploiting the fact that solution multiplicity typically varies locally rather than

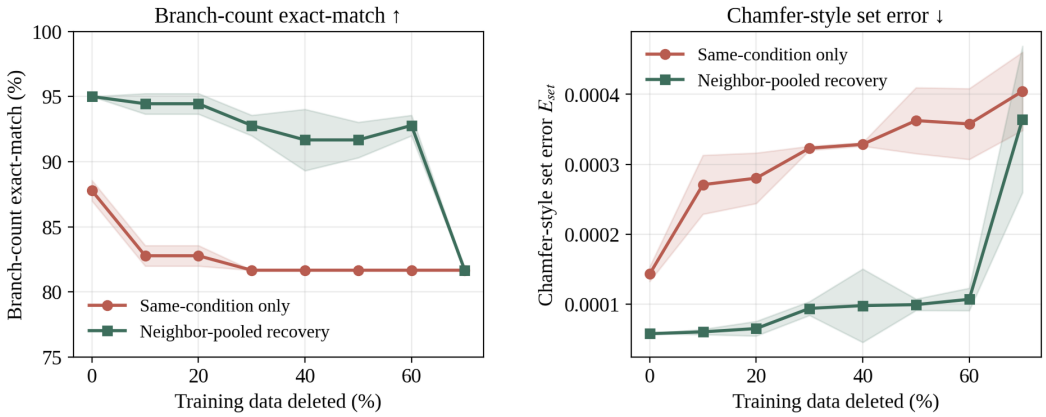

Figure 21: **Robustness of target solution-count estimation to incomplete training observations.** On the continuum-robot dataset with an FNO backbone, we randomly delete individual training observations and recompute the target solution count M_i either from observations at the same condition only or by pooling each condition with its five nearest neighbors before reclustering. The multi-solution predictor is fixed and only the activation model is retrained. Neighbor-pooled estimation preserves higher branch-count exact-match accuracy and lower symmetric set error over moderate deletion rates, remaining robust through 60% data removal. Both methods degrade at the extreme 70% deletion level, where too few training observations remain for reliable recovery. Curves show mean $\pm$ one standard deviation over three independent deletion seeds.

independently from one condition to the next. To evaluate the robustness of this estimate, we randomly delete 0–70% of individual training observations on the continuum-robot dataset with the FNO backbone and recompute M_i from scratch at each deletion level, either using only surviving same-condition observations or pooling each condition with its five nearest neighbors before reclustering. The multi-solution predictor is kept fixed and only the activation model is retrained, while evaluation is performed on the same untouched test set using branch counts computed from the complete data as reference. As shown in Fig. 21, neighbor-pooled estimation maintains approximately 91–95% branch-count exact-match accuracy through 60% deletion and yields substantially lower symmetric set error, whereas same-condition estimation degrades much earlier. This suggests that borrowing nearby observations provides a more reliable estimate of solution multiplicity when the training data are incomplete.

A.9 Analysis of Solution Specialization under Multi-Solution Data

We provide a simplified analysis to illustrate two benefits of the proposed solution features: avoiding the averaging behavior of deterministic regression and improving the efficiency of covering multiple solutions compared with independent probabilistic sampling.

Setup. Consider a fixed physical condition $a \in \mathcal{A}$. Suppose that a admits M distinct solutions,

$$\mathcal{S}(a) = \left\{ u^{(1)}(a), \ldots, u^{(M)}(a) \right\}, \tag{38}$$

where $u^{(m)}(a) \in \mathcal{U}$ denotes the m-th valid solution. The training data reveal only one solution at a time. We denote the probability of observing solution $u^{(m)}(a)$ by

$$\pi_m(a) = \Pr\left(U = u^{(m)}(a) \mid a \right), \qquad \sum_{m=1}^{M} \pi_m(a) = 1. \tag{39}$$

Here, U is a random variable representing the solution observed in the training data for condition a.

Our model introduces K learned solution features $\{z_k\}_{k=1}^{K}$, where $K \geq M$, and produces one candidate solution from each feature,

$$\hat{u}_k(a) = F_\theta(a, z_k), \qquad k = 1, \ldots, K. \tag{40}$$

For simplicity, we first consider deterministic candidates; the same argument extends naturally to stochastic candidates.

Deterministic regression averages distinct solutions. A conventional deterministic simulator predicts a single solution $f(a)$. Under the squared reconstruction loss,

$$\mathcal{R}_{\mathrm{det}}(f \mid a) = \mathbb{E}\left[\|f(a) - U\|_2^2 \mid a\right], \tag{41}$$

the population-optimal prediction is the conditional mean,

$$f^*(a) = \mathbb{E}[U \mid a] = \sum_{m=1}^{M} \pi_m(a) u^{(m)}(a). \tag{42}$$

Its minimum expected loss is

$$\mathcal{R}^*_{\mathrm{det}}(a) = \sum_{m=1}^{M} \pi_m(a) \left\| u^{(m)}(a) - f^*(a) \right\|_2^2. \tag{43}$$

Therefore, whenever multiple distinct solutions occur with nonzero probability, $\mathcal{R}^*_{\mathrm{det}}(a) > 0$. Moreover, $f^*(a)$ may lie between the valid solutions and need not itself belong to $\mathcal{S}(a)$.

Best-of-K solution features. Our training objective instead allows several candidate predictions and matches each observed solution to the candidate that explains it best:

$$\mathcal{L}_K(a, U) = \min_{k \in \{1,\dots,K\}} \ell(\hat{u}_k(a), U). \tag{44}$$

Suppose that the learned solution features specialize such that every valid solution is represented by at least one candidate,

$$\forall m \in \{1, \dots, M\}, \qquad \exists k_m \quad \text{such that} \quad \hat{u}_{k_m}(a) = u^{(m)}(a). \tag{45}$$

Since $K \geq M$, such an assignment is possible. Under exact specialization,

$$\mathbb{E}\left[\mathcal{L}_K(a, U) \mid a\right] = 0. \tag{46}$$

Thus, unlike a single deterministic predictor, the best-of-K objective can represent all distinct solutions without averaging them.

Comparison with independent probabilistic sampling. We next consider a generative simulator that correctly reproduces the conditional data distribution. Suppose we draw K independent predictions

$$\tilde{U}_1, \dots, \tilde{U}_K \overset{\text{i.i.d.}}{\sim} p_{\mathrm{data}}(u \mid a). \tag{47}$$

For a particular solution $u^{(m)}(a)$, each draw produces this solution with probability $\pi_m(a)$. Therefore, the probability that none of the K samples produces $u^{(m)}(a)$ is

$$P^{(m)}_{\mathrm{miss}}(K) = (1 - \pi_m(a))^K. \tag{48}$$

Equivalently, its probability of being covered by at least one sample is

$$P^{(m)}_{\mathrm{cover}}(K) = 1 - (1 - \pi_m(a))^K. \tag{49}$$

If we measure best-of-K prediction using the zero–one loss, where the loss is zero if one candidate exactly matches the observed solution and one otherwise, the expected loss of independent sampling is

$$\mathbb{E}\left[\mathcal{L}^{\mathrm{iid}}_K \mid a\right] = \sum_{m=1}^{M} \pi_m(a) \left(1 - \pi_m(a)\right)^K. \tag{50}$$

For any genuinely multi-solution distribution and finite K, this quantity is strictly positive.

In contrast, under the specialization condition in Eq. equation 45, querying the K learned solution features deterministically covers every solution represented by the model, giving

$$\mathbb{E}\left[\mathcal{L}^{\mathrm{spec}}_K \mid a\right] = 0. \tag{51}$$

This difference becomes particularly important for rare solutions. To observe solution $u^{(m)}(a)$ with probability at least $1-\delta$ using independent sampling requires

$$K \geq \frac{\log \delta}{\log(1-\pi_m(a))} \approx \frac{1}{\pi_m(a)} \log \frac{1}{\delta}, \tag{52}$$

where the approximation holds for small $\pi_m(a)$. Hence, the sampling cost grows inversely with the frequency of a rare solution. Once solution features specialize, however, the corresponding solution can be accessed directly through its feature rather than rediscovered through random sampling.

Stochastic solution features. A solution-conditioned generative model may retain stochasticity. Let

$$q_{km}(a) = \Pr\left(\hat{U}_k = u^{(m)}(a) \mid a, z_k\right), \tag{53}$$

where $\hat{U}_k$ denotes a stochastic prediction conditioned on solution feature z_k. Querying each of the K hypotheses once misses solution mode m with probability

$$P^{(m)}_{\text{miss,ours}} = \prod_{k=1}^{K} \left(1 - q_{km}(a)\right). \tag{54}$$

For comparison, assume K independent baseline samples have the same marginal probability

$$\pi_m(a) = \frac{1}{K} \sum_{k=1}^{K} q_{km}(a) \tag{55}$$

of producing mode m. Their miss probability is

$$P^{(m)}_{\text{miss,base}} = \left(1 - \pi_m(a)\right)^K. \tag{56}$$

By the arithmetic–geometric mean inequality,

$$\prod_{k=1}^{K} \left(1 - q_{km}(a)\right) \leq \left(1 - \frac{1}{K} \sum_{k=1}^{K} q_{km}(a)\right)^K = \left(1 - \pi_m(a)\right)^K. \tag{57}$$

Hence, for fixed marginal mode probability, distributing probability mass unevenly across addressable hypotheses can only reduce the probability of missing that mode. Equality holds when all hypotheses reproduce the same distribution, while specialization makes the inequality strict.

This result characterizes the benefit of specialization once it is learned; it does not imply that convergence of the training objective alone guarantees such specialization. Our best-match training is designed to encourage this behavior, which we evaluate empirically through branch specialization and sampling-efficiency experiments.