NEURAL HOMOTOPY SYSTEM: DIFFERENTIABLE PREDICTOR-CORRECTOR FOR HOMOTOPY PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

The homotopy paradigm provides a general strategy for solving difficult problems. It connects a hard target problem to a simple source problem through a continuous interpolation, then tracks the resulting solution trajectory, typically with a predictor-corrector algorithm. This paradigm underlies methods across many domains, but the interpolation is usually hand-designed and task-specific. Existing learning-based attempts to replace it remain instance-specific, requiring retraining for every new problem instance. We propose **Neural Homotopy System (NHS)**, a differentiable predictor-corrector framework in which the predictor and corrector are learned neural modules, trained once and amortized over problem instances within a given problem family. NHS parameterizes the predictor with a neural ordinary differential equation that learns the solution trajectory across homotopy levels, while the corrector uses a deep equilibrium model that drives the predicted solution toward a problem-specific equilibrium condition. A predictor-anchored proximal term makes this equilibrium explicitly depend on the prediction, allowing gradients to propagate across homotopy levels and enabling end-to-end training of the complete trajectory. Instantiating this shared architecture separately for unconstrained optimization, constrained optimization, and conditional probabilistic sampling, NHS consistently achieves higher solution accuracy than classical local and hand-designed homotopy methods, as well as existing learning-based baselines, across all three settings.

1 INTRODUCTION

The homotopy paradigm provides a general strategy for solving difficult problems and has been widely adopted across different domains under various names. Representative examples include graduated optimization for global optimization (Mobahi & Fisher III, 2015), graduated non-convexity for robust optimization (Yang et al., 2020), interior-point methods for constrained optimization (Monteiro & Adler, 1989), annealed importance sampling for probabilistic inference and sampling (Hastings, 1970; Neal, 2001), and homotopy continuation methods for root finding (Bates et al., 2013). The underlying idea is intuitive. A difficult target problem is connected to a simple source problem through a continuous homotopy interpolation, and the solution trajectory is then tracked from the simple solution toward the target. Practical solvers for these problems typically follow a predictor-corrector structure, where a predictor advances along the homotopy interpolation and a corrector refines the prediction at each level (Allgower & Georg, 2012). However, the homotopy interpolation itself is generally specified by hand-designed numerical rules (Yang et al., 2020; Chehab et al., 2023), which are typically suboptimal and highly task-specific.

Learning has recently been applied to parts of this pipeline in two distinct ways. One line of work learns the homotopy interpolation itself rather than hand-designing it (Máté & Fleuret, 2023), but remains largely instance-specific and requires retraining when applied to a new problem instance, since the learned interpolation is tied to a single fixed target. A separate line of work instead keeps the numerical predictor-corrector solver fixed and learns how to run it. NPC (Mai et al., 2026) casts homotopy solving across several domains as a sequential decision process and learns predictor step sizes and corrector termination policies with reinforcement learning. This improves efficiency but leaves the predictor and corrector themselves hand-designed, which does not permit gradient-based training through the solution trajectory (Mai et al., 2026).

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.

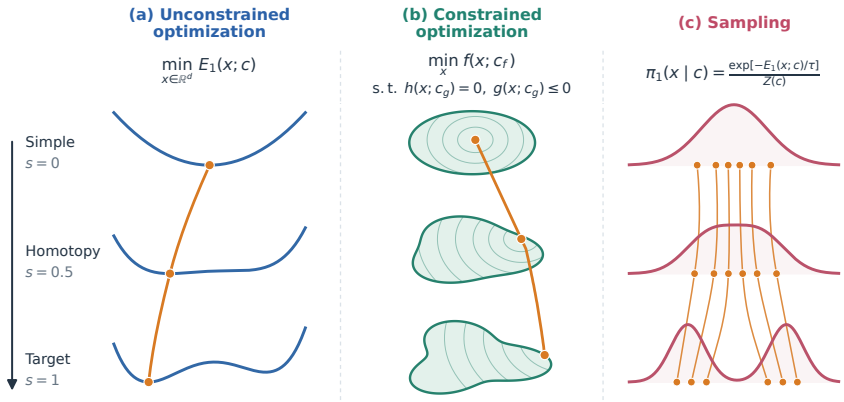

Figure 1: Homotopy view of three application domains: unconstrained optimization, constrained optimization, and probabilistic sampling. Homotopy interpolation transforms a difficult problem into a sequence of easier problems, while the solver tracks the corresponding implicit solution trajectory to the target problem. Our method aims to learn this homotopy interpolation within a unified predictor-corrector framework.

We instead ask whether the predictor and corrector can themselves be replaced by learned, differentiable modules, so the entire system is trained end-to-end with gradients rather than by tuning a policy around a fixed solver. We propose **Neural Homotopy System (NHS)**, which parameterizes the predictor as a neural ordinary differential equation (Neural ODE) (Chen et al., 2018) that learns the solution trajectory across homotopy levels, and the corrector as a deep equilibrium model (DEQ) (Bai et al., 2019) that drives the predicted solution toward a problem-specific equilibrium condition. A predictor-anchored proximal term couples the two components. It makes the corrector's equilibrium explicitly depend on the predictor's proposal, allowing implicit gradients to flow from the corrector back through the predictor and adjoint gradients to flow across homotopy levels, so the complete trajectory can be trained jointly.

NHS shares this predictor-corrector architecture across problem classes, instantiating it separately for each one with a problem-specific state representation, equilibrium condition, and training objective, while amortizing training over problem instances within that class. We instantiate NHS for unconstrained optimization, constrained optimization, and conditional probabilistic sampling. Across all three, NHS generalizes to unseen problem instances without per-instance retraining and consistently achieves higher solution accuracy than hand-designed homotopy and gradient-free smoothing methods, as well as existing learning-based baselines.

In summary, our **contributions** are as follows:

- A differentiable homotopy system that replaces the predictor and corrector with a Neural ODE and a DEQ, respectively.
- A predictor-anchored proximal term that makes the corrected equilibrium explicitly depend on the prediction, enabling gradient propagation across homotopy levels and end-to-end training through the complete trajectory.
- Problem-specific instantiations of this shared architecture, with corresponding state representations, equilibrium conditions, and training objectives, for unconstrained optimization, constrained optimization, and conditional probabilistic sampling, each amortized over problem instances within its family.

2 Related Work

Homotopy predictor-corrector methods appear across many problem domains, but learning-based improvements to them have developed largely independently across optimization, constrained optimization, and sampling. We review them here and highlight the gap that motivates our work. A full discussion is provided in Appendix A.

Learning within Predictor-Corrector Solvers. Prior work has learned individual numerical components of a predictor-corrector scheme while leaving the scheme itself fixed, such as ODE integrators for pretrained generative flows (Shaul et al., 2024) or fixed-point initializations for DEQ inference (Bai et al., 2021), alongside training-free high-order solvers for diffusion sampling (Zhao et al., 2023). Closer to our setting, NPC formulates homotopy algorithms across several domains as a shared decision process and learns predictor step sizes and corrector termination policies while retaining the underlying numerical state updates (Mai et al., 2026).

Learning Homotopy Paths per Problem Class. Within individual problem classes, learned optimizers and diverse-initialization networks amortize part of the optimization process (Andrychowicz et al., 2016; Sharony et al., 2024), while Continuation Path Learning learns a level-conditioned network for a single homotopy path (Lin et al., 2023). Learned constrained-optimization methods either penalize constraints during training (Fioretto et al., 2020; Park & Van Hentenryck, 2023) or explicitly repair predictions to restore feasibility (Donti et al., 2021; Nguyen & Donti, 2026; Liang et al., 2024; Lüken & Lucia, 2026). In sampling, learned approaches adapt the density path, the transport between its levels, or both (Máté & Fleuret, 2023; Maurais et al., 2025; Arbel et al., 2021; Matthews et al., 2022; Zhang et al., 2021).

Across these settings, the predictor and corrector are either fixed numerical procedures wrapped by a learned policy or component, or are learned for a single problem instance rather than amortized across a problem family. NHS instead replaces both the predictor and corrector with differentiable neural modules, trained end-to-end and amortized across problem instances within each family.

3 Background

3.1 Homotopy predictor-corrector methods

Homotopy predictor-corrector methods track a solution path from an easy source problem $\mathcal{P}_0$ to a difficult target problem $\mathcal{P}_1$ through a continuous family $\{\mathcal{P}_s\}_{s\in[0,1]}$. At scheduled levels $0 = s_0 < \cdots < s_L = 1$, the state u_ℓ approximates the solution at s_ℓ, as either an optimization iterate or a particle representation of a probability distribution. Starting from u_0, each step applies:

$$\widehat{u}_\ell = \mathrm{Pred}(u_{\ell-1}, s_{\ell-1}, s_\ell), \qquad u_\ell = \mathrm{Corr}(\widehat{u}_\ell, s_\ell), \quad \ell = 1, \ldots, L. \tag{1}$$

The predictor Pred uses the previous state and the change in homotopy level to construct the predicted state $\widehat{u}_\ell$, while the corrector Corr refines this prediction for the new problem $\mathcal{P}_{s_\ell}$, yielding the updated state u_ℓ. These steps alternate until the target level $s_L = 1$ is reached.

3.2 Neural ordinary differential equation (ODE) and adjoint differentiation

A Neural ODE parameterizes a continuous transformation from state z_a to z_b by:

$$\frac{dz}{ds} = f_\phi(z, s), \quad z(s_a) = z_a, \quad z_b := \mathrm{ODESolve}_\phi(z_a, s_a, s_b) = z_a + \int_{s_a}^{s_b} f_\phi(z(s), s)\, ds. \tag{2}$$

For a scalar loss $\mathcal{L}(z_b)$, the continuous adjoint $a(s) = \nabla_{z(s)}\mathcal{L}$ is integrated backward from s_b to s_a, yielding $\nabla_{z_a}\mathcal{L} = a(s_a)$, and satisfies (Chen et al., 2018):

$$\frac{da}{ds} = -\left(\frac{\partial f_\phi}{\partial z}\right)^\top a, \quad a(s_b) = \nabla_{z_b}\mathcal{L}, \quad \nabla_{z_a}\mathcal{L} = a(s_a) = \nabla_{z_b}\mathcal{L} + \int_{s_b}^{s_a} -\left(\frac{\partial f_\phi}{\partial z}\right)^\top a(s)\, ds. \tag{3}$$

3.3 Deep equilibrium model (DEQ) and implicit differentiation

A DEQ defines its output as a fixed point $z^\star = g_\theta(z^\star, \omega)$, or equivalently as the root of $G_\theta(z^\star, \omega) = 0$, where ω collects parameters and inputs. Given an initial state, the forward pass first solves this fixed-point equation to obtain $z^\star$. Once the equilibrium is found, its sensitivity with respect to ω can be computed directly by implicit differentiation, without differentiating through the fixed-point iterations (Bai et al., 2019):

$$\frac{dz^\star}{d\omega} = -\left(\frac{\partial G}{\partial z}\right)^{-1} \frac{\partial G}{\partial \omega}. \tag{4}$$

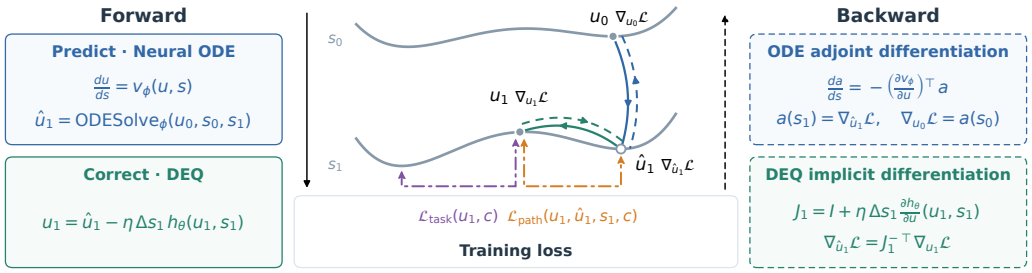

Figure 2: A Neural ODE predicts $\hat{u}_1$ from u_0, and a DEQ corrects it to u_1. Solid and dashed trajectories depict forward tracking and backward differentiation, respectively. Task and path losses jointly train the solver for solution quality and prediction-correction consistency.

4 Neural Homotopy System

In the predictor-corrector framework introduced in Sec. 3, the homotopy interpolation, predictor, and corrector are prescribed components. We instead make the entire homotopy system learnable and differentiable, allowing these components to be jointly optimized end-to-end to learn an effective homotopy interpolation and its associated solution trajectory. We introduce the learnable predictor and corrector as a Neural ODE and a DEQ, respectively, in Sec. 4.1. Sec. 4.2 assembles these components into the complete NHS trajectory and derives the backward differentiation through the coupled predictor-corrector system for end-to-end training. Moreover, NHS is trained in an amortized fashion over problem instances, and the same algorithmic framework applies across different problem classes as detailed in Sec. 5.

4.1 Predictor and Corrector

Predictor as Neural ODE. Given the corrected state $u_{\ell-1}$, a conditional Neural ODE predicts its state at the next level:

$$\frac{du}{ds} = v_\phi(u, s), \qquad \widehat{u}_\ell = \mathrm{ODESolve}_\phi(u_{\ell-1}, s_{\ell-1}, s_\ell). \tag{5}$$

Corrector as DEQ. At level s_ℓ, let $h_\theta(\cdot, s_\ell)$ denote a problem-specific correction operator. The corrector is the equilibrium of the predictor-anchored map:

$$u_\ell = \widehat{u}_\ell - \alpha_\ell h_\theta(u_\ell, s_\ell), \qquad \alpha_\ell = \eta \Delta s_\ell. \tag{6}$$

We approximate this map by damped fixed-point iteration with a stopping tolerance. For $\alpha_\ell > 0$, this fixed point is equivalently the proximal equilibrium:

$$h_\theta(u, s_\ell) + \frac{u - \widehat{u}_\ell}{\alpha_\ell} = 0. \tag{7}$$

The second term is the proximal anchor: it favors states near the predictor proposal, but does not guarantee preservation of a nonconvex solution branch. It also makes the equilibrium explicitly depend on the predicted state, allowing implicit gradients to propagate through the corrector to the predictor. If $h_\theta = \nabla_u \mathcal{H}_\theta$ is conservative, Eq. (7) is the stationarity condition of:

$$\mathcal{H}_\theta(u, s_\ell) + \frac{\|u - \widehat{u}_\ell\|^2}{2\alpha_\ell}. \tag{8}$$

4.2 Forward Tracking and Backward Differentiation

Having defined the essential components of NHS, we now assemble them into a complete differentiable predictor-corrector algorithm, as illustrated in Fig. 2. The forward pass tracks the solution trajectory through successive prediction and correction steps, while the backward pass propagates training signals through these coupled computations to jointly optimize the two components.

Forward Tracking. Starting from an initial state u_0 associated with the source problem, NHS advances through the fixed homotopy levels $s_0, \ldots, s_L$. At each level s_ℓ, the Neural ODE-based predictor (Eq. (5)) transports the previous corrected state $u_{\ell-1}$ across the interval $[s_{\ell-1}, s_\ell]$, producing a proposal $\widehat{u}_\ell$ for the next problem. The DEQ-based corrector (Eq. (6)) then refines this proposal by solving the predictor-anchored equilibrium defined above. The corrected state u_ℓ initializes the next prediction, producing the coupled trajectory:

$$u_{\ell-1} \xrightarrow{\mathrm{ODE}_\phi} \widehat{u}_\ell \xrightarrow{\mathrm{DEQ}_\theta} u_\ell. \tag{9}$$

Loss. We jointly train the predictor and corrector over a distribution of instances within a problem family $c \sim p(c)$, using both the terminal output and the intermediate states:

$$\min_{\phi,\theta} \mathbb{E}_c \left[\mathcal{L}_{\mathrm{task}}(u_L, c) + \sum_{\ell=1}^{L} \lambda_\ell \mathcal{L}_{\mathrm{path}}(u_\ell, \widehat{u}_\ell, s_\ell, c) \right], \tag{10}$$

where λ_ℓ is the fixed loss weight. The task loss evaluates the quality of the terminal output for the target problem, such as its objective value in optimization or a distributional discrepancy in sampling. The path loss encourages consistency between predictor proposals and corrected states. Their application-specific forms are detailed in Sec. 5. Ideally, when the corrected state u_ℓ coincides with the predictor proposal $\widehat{u}_\ell$, the proximal term vanishes, indicating that the predictor itself tracks the homotopy solution trajectory.

Backward Differentiation. Let $\mathcal{L}$ denote the training loss at homotopy level ℓ. We propagate the gradients backward from $\nabla_{u_\ell}\mathcal{L}$ through $\nabla_{\widehat{u}_\ell}\mathcal{L}$ to $\nabla_{u_{\ell-1}}\mathcal{L}$. At each level, gradients first pass through the corrector to the predictor proposal, and then through the predictor to the previous corrected state:

$$\nabla_{u_\ell}\mathcal{L} \xrightarrow[\text{Implicit differentiation}]{\left(\frac{\partial u_\ell}{\partial \widehat{u}_\ell}\right)^\top} \nabla_{\widehat{u}_\ell}\mathcal{L} \xrightarrow[\text{Adjoint differentiation}]{\left(\frac{\partial \widehat{u}_\ell}{\partial u_{\ell-1}}\right)^\top} \nabla_{u_{\ell-1}}\mathcal{L}. \tag{11}$$

The corrector and predictor backward passes also contribute to θ and ϕ, respectively. We detail the two backward steps below.

Differentiating through the corrector. At level ℓ, the corrected state u_ℓ is defined by the predictor-anchored equilibrium (Eq. (6)). Define the Jacobian of the equilibrium equation with respect to the corrected state:

$$J_\ell = I + \alpha_\ell \frac{\partial h_\theta}{\partial u}(u_\ell, s_\ell). \tag{12}$$

At a converged equilibrium on a differentiable solution branch, implicit differentiation (Eq. (4)) gives:

$$\frac{\partial u_\ell}{\partial \widehat{u}_\ell} = J_\ell^{-1}, \qquad \frac{\partial u_\ell}{\partial \theta} = -\alpha_\ell J_\ell^{-1} \frac{\partial h_\theta}{\partial \theta}. \tag{13}$$

The first expression describes how the corrected state responds to changes in the prediction, while the second describes its response to changes in the corrector parameters. Applying the chain rule then accumulates gradients at the predictor proposal and the shared corrector parameters:

$$\nabla_{\widehat{u}_\ell}\mathcal{L} = \left(\frac{\partial u_\ell}{\partial \widehat{u}_\ell}\right)^\top \nabla_{u_\ell}\mathcal{L}, \qquad \nabla_\theta \mathcal{L} = \left(\frac{\partial u_\ell}{\partial \theta}\right)^\top \nabla_{u_\ell}\mathcal{L}. \tag{14}$$

Thus, the corrector backward pass contributes to training its own parameters and passes the downstream learning signal to the predictor.

Differentiating through the predictor. Next, we propagate the accumulated proposal gradient $\nabla_{\widehat{u}_\ell}\mathcal{L}$ through the Neural ODE to the previous level. Using this gradient as the terminal condition for the adjoint $a(s_\ell)$, we integrate backward from s_ℓ to $s_{\ell-1}$:

$$a(s_\ell) = \nabla_{\widehat{u}_\ell}\mathcal{L}, \qquad \frac{da}{ds} = -\left(\frac{\partial v_\phi}{\partial u}\right)^\top a. \tag{15}$$

This yields contributions to the previous corrected state and the shared predictor parameters:

$$\nabla_{u_{\ell-1}}\mathcal{L} = a(s_{\ell-1}), \qquad \nabla_\phi \mathcal{L} = \int_{s_{\ell-1}}^{s_\ell} \left(\frac{\partial v_\phi}{\partial \phi}\right)^\top a(s)\, ds. \tag{16}$$

All partial derivatives are evaluated along the forward ODE trajectory of the current segment. The adjoint is integrated backward, while the parameter-gradient integral is written with increasing integration limits.

5 APPLICATIONS

We instantiate NHS for unconstrained optimization, constrained optimization, and conditional probabilistic sampling. All applications share the Neural ODE-based predictor and the differentiable predictor–corrector framework introduced above, while adapting the state representation, homotopy parameterization, corrector realization, and terminal objective. The practical implementation of the corrector differs across problem classes; see Appendices B to E for details.

5.1 UNCONSTRAINED OPTIMIZATION

Problem Formulation. For a problem instance specified by instance c, we seek:

$$x^\star(c) \in \arg\min_{x\in\mathbb{R}^d} E_1(x;c), \tag{17}$$

where E_1 is the target objective. The solver state is $u = x \in \mathbb{R}^d$.

Homotopy Parameterization. Starting from an easy objective E_0, we construct an endpoint-preserving operator homotopy:

$$h_\theta(x,s;c) = (1-s)\nabla_x E_0(x;c) + s\nabla_x E_1(x;c) + s(1-s)r_\theta(x,s,c). \tag{18}$$

where the factor $s(1-s)$ preserves the endpoint operators while allowing the residual network r_θ to modify intermediate levels. Since r_θ needs not be conservative, the corrector is defined through an operator equilibrium rather than the minimization of a learned scalar energy.

Corrector. Given the predictor proposal $\widehat{x}_\ell$, the corrected state satisfies the predictor-anchored proximal equilibrium:

$$h_\theta(x_\ell, s_\ell; c) + \frac{x_\ell - \widehat{x}_\ell}{\alpha_\ell} = 0. \tag{19}$$

Training Loss. We combine supervised endpoint loss, the terminal objective value, and prediction–correction consistency along the trajectory:

$$\mathcal{L}_{\text{opt}} = \mathbb{E}_c\left[\lambda_{\text{sup}}\mathcal{L}_{\text{sup}}\big(x_L, x_{\text{ref}}(c)\big) + \lambda_{\text{obj}}E_1(x_L;c) + \frac{\lambda_{\text{cons}}}{L}\sum_{\ell=1}^{L}\frac{\|\widehat{x}_\ell - \text{sg}(x_\ell)\|_2^2}{d\,\Delta s_\ell}\right]. \tag{20}$$

Here $x_{\text{ref}}(c)$ denotes a reference solution when available, and $\mathcal{L}_{\text{sup}}$ is a task-specific supervised endpoint loss, instantiated as Smooth-L1 or MSE in our experiments. The objective term directly measures solution quality under the target objective, while the path-consistency term encourages the predictor to match the corrected states. The exact loss weights and additional task-specific regularization terms used in the reported experiments are given in Appendix B.

5.2 CONSTRAINED OPTIMIZATION

Problem Formulation. For context $c = (c_f, c_g)$, we consider:

$$\min_x f(x;c_f) \quad \text{s.t.} \quad h(x;c_g) = 0, \qquad g(x;c_g) \le 0, \tag{21}$$

where $h : \mathbb{R}^d \to \mathbb{R}^p$ and $g : \mathbb{R}^d \to \mathbb{R}^q$. We lift the state to the primal-dual variable:

$$z = (x, \nu, \lambda) \in D := \mathbb{R}^d \times \mathbb{R}^p \times \mathbb{R}^q_+. \tag{22}$$

Homotopy Parameterization. The target KKT operator is:

$$A_1(z;c) = \begin{bmatrix} \nabla_x f(x;c_f) + J_h(x;c_g)^\top \nu + J_g(x;c_g)^\top \lambda \\ -h(x;c_g) \\ -g(x;c_g) \end{bmatrix}, \tag{23}$$

with the corresponding equilibrium condition:

$$0 \in A_1(z;c) + N_D(z), \tag{24}$$

where N_D is the normal cone of D. This inclusion encodes stationarity, primal and dual feasibility, and complementarity. Given an easy source operator, such as $A_0(z) = (\nabla_x E_0(x), \nu, \lambda)$, we parameterize the homotopy as:

$$A_{\theta,s}(z;c) = (1-s)A_0(z) + sA_1(z;c) + s(1-s)a_\theta(z,s,c). \tag{25}$$

The neural ODE predictor acts on the full primal-dual state, transporting the primal variables and their associated multipliers together.

Corrector. Given the predicted state $\widehat{z}_\ell$, the proximal corrector satisfies:

$$0 \in A_{\theta,s_\ell}(z_\ell;c) + N_D(z_\ell) + \frac{z_\ell - \widehat{z}_\ell}{\alpha_\ell}. \tag{26}$$

Training Loss. Using supervised endpoint loss, the terminal objective value, and the consistency of the primal-dual trajectory, we write:

$$\mathcal{L}_{\text{copt}} = \mathbb{E}_c\left[\lambda_{\text{sup}}\mathcal{L}_{\text{sup}}\big(x_L, x_{\text{ref}}(c)\big) + \lambda_{\text{obj}} f(x_L;c_f) + \frac{\lambda_{\text{cons}}}{L}\sum_{\ell=1}^{L}\frac{\|\widehat{z}_\ell - \text{sg}(z_\ell)\|_2^2}{(d+p+q)\,\Delta s_\ell}\right]. \tag{27}$$

Here $x_{\text{ref}}(c)$ denotes a reference primal solution when available, and $\mathcal{L}_{\text{sup}}$ is instantiated according to the task, *e.g.*, as MSE in the constrained-optimization experiments. The objective term measures terminal solution quality, while the consistency term aligns the predicted and corrected primal-dual states. The exact task-specific losses and weights used in Sec. 6.2 are reported in Appendix C.

5.3 Conditional Probabilistic Sampling

Problem Formulation. Given a conditional target distribution:

$$\pi_1(x \mid c) = \frac{\exp[-E_1(x;c)/\tau]}{Z(c)}, \tag{28}$$

we learn a problem-conditioned solver that transports an easily sampled source distribution $\rho_0(x) \propto \exp[-E_0(x)/\tau]$ toward $\pi_1(\cdot \mid c)$. The particle state is $X = (x_1, \ldots, x_N) \in \mathbb{R}^{N\times d}$.

Homotopy Parameterization. We construct an endpoint-preserving energy path:

$$\begin{aligned} &H_{\theta,s}(x;c) = (1-s)E_0(x) + sE_1(x;c) + s(1-s)V_\theta(x,s,c), \\ &\pi_{\theta,s}(x \mid c) \propto \exp[-H_{\theta,s}(x;c)/\tau]. \end{aligned} \tag{29}$$

All intermediate distributions are assumed to be normalizable and have finite second moments. At each homotopy level, the Neural ODE-based predictor pushes the previous distribution forward $\widehat{\rho}_\ell = (P_{\phi,\ell})_{\#}\rho_{\ell-1}$, where $P_{\phi,\ell}$ denotes the learned flow from $s_{\ell-1}$ to s_ℓ.

Corrector. Given the predicted distribution $\widehat{\rho}_\ell$, we perform the Wasserstein proximal correction

$$\rho_\ell \in \arg\min_{\rho}\left\{\mathbb{E}_{x\sim\rho}\big[H_{\theta,s_\ell}(x;c)\big] + \tau\int \rho(x)\log\rho(x)\,dx + \frac{W_2^2(\rho,\widehat{\rho}_\ell)}{2\alpha_\ell}\right\}. \tag{30}$$

The first two terms define the free energy and drive relaxation toward the current homotopy target, while the Wasserstein term anchors the update to the prediction in distribution space rather than through a particle-wise quadratic potential.

Training Loss. We use a terminal Wasserstein loss and a Wasserstein path-consistency loss:

$$\mathcal{L}_{\text{samp}} = \mathbb{E}_{c,\zeta}\left[\lambda_{\text{dist}} W_2^2(\rho_L, \pi_1(\cdot \mid c)) + \frac{\lambda_{\text{cons}}}{L}\sum_{\ell=1}^{L}\frac{W_2^2(\widehat{\rho}_\ell, \text{sg}(\rho_\ell))}{d\,\Delta s_\ell}\right], \tag{31}$$

where ζ collects the initial particles and any reparameterized correction noise. The terminal term measures the discrepancy between the final and target distributions, while path consistency encourages agreement between predicted and corrected distributions. In the particle representation, $\text{sg}(\rho_\ell)$ means detaching the corrected particle positions.

Table 1: Performance on unconstrained optimization problems.

Method	Hard Ackley		Hard Rastrigin		Switching Himmelblau	
	Mean distance to reference ↓	Recall (%) ↑ L2 ≤ 0.05	Mean distance to reference ↓	Recall (%) ↑ L2 ≤ 0.01	Mean distance to reference ↓	Recall (%) ↑ L2 ≤ 0.1
L-BFGS-B	1.1335	29.69	1.9072	7.03	4.1941	30.47
GD	1.6135	7.81	1.2958	5.47	4.1941	30.47
GH	0.0647 ± 0.0021	39.53 ± 5.08	0.6261 ± 0.0204	0.94 ± 0.65	3.5537 ± 0.0922	27.97 ± 3.51
EPGS	0.0277 ± 0.0013	92.19 ± 2.07	0.2173 ± 0.0237	1.25 ± 0.70	3.9024 ± 0.0501	25.78 ± 1.75
PGS	0.0461 ± 0.0041	71.72 ± 4.15	0.1804 ± 0.0163	0.16 ± 0.35	3.8034 ± 0.0733	31.09 ± 3.10
ZOSLGH-d	0.3665 ± 0.0153	56.88 ± 1.50	1.7411 ± 0.0020	0.78 ± 0.00	3.6927 ± 0.1364	37.97 ± 2.25
ZOSLGH-r	0.0476 ± 0.0061	72.66 ± 6.08	1.0708 ± 0.0145	6.25 ± 1.24	4.0931 ± 0.3564	37.19 ± 5.23
MISO PD $K = 4$	0.0222 ± 0.0042	92.66 ± 2.95	0.0208 ± 0.0104	28.59 ± 13.72	0.3344 ± 0.0368	32.03 ± 7.61
MISO WTA $K = 4$	0.0464 ± 0.0144	62.19 ± 21.36	0.0531 ± 0.0117	4.38 ± 1.71	0.2432 ± 0.1491	50.16 ± 31.19
Conditional CPL	0.0343 ± 0.0121	80.78 ± 25.03	0.5692 ± 0.4777	1.09 ± 1.71	3.2330 ± 0.0317	38.59 ± 16.00
NHS (ours)	**0.0175 ± 0.0017**	**95.78 ± 2.38**	**0.0012 ± 0.0002**	**100.00 ± 0.00**	**0.0291 ± 0.0076**	**95.00 ± 1.42**

Table 2: Performance on constrained optimization problems.

Method	Eq. violation ↓		Ineq. violation ↓		Mean objective difference to reference ↓	Mean distance to reference ↓
	Mean	Max	Mean	Max		
Nonconvex SOCP						
FSNet	9.640×10^{-5}	1.086×10^{-3}	2.804×10^{-7}	5.962×10^{-5}	4.84576	15.25246
DC3	$\mathbf{2.122 \times 10^{-13}}$	$\mathbf{2.786 \times 10^{-13}}$	8.259×10^{-2}	3.939×10^{0}	7.33032	16.14036
NHS	3.249×10^{-13}	5.464×10^{-13}	**0**	**0**	**−0.94791**	**12.87868**
Nonconvex QCQP						
FSNet	1.073×10^{-4}	8.659×10^{-4}	2.211×10^{-6}	7.923×10^{-5}	−0.25441	5.67941
DC3	$\mathbf{1.996 \times 10^{-13}}$	$\mathbf{2.693 \times 10^{-13}}$	1.634×10^{-1}	1.132×10^{0}	2.84710	11.66604
NHS	2.764×10^{-13}	3.822×10^{-13}	**0**	**0**	**−0.28020**	**3.88069**

6 EXPERIMENTS

In all tables, **bold** and underlined entries indicate the **best** and second-best results for each metric, respectively. Problem definitions, visualizations of problem landscapes and solver trajectories, implementation details, experimental settings, evaluation protocols, and metric definitions are provided in Appendices B to G.

6.1 UNCONSTRAINED GLOBAL OPTIMIZATION

We evaluate NHS on three two-dimensional, context-dependent nonconvex optimization benchmarks: Hard Ackley (Ackley, 2012), Hard Rastrigin (Rastrigin, 1974), and Switching Himmelblau (Himmelblau et al., 2018). We compare against gradient-based methods (GD and L-BFGS-B (Byrd et al., 1995)), gradient-free smoothing methods (GH (Mobahi & Fisher III, 2015), EPGS and PGS (Xu, 2024), and ZOSLGH-d/r (Iwakiri et al., 2022)), and learning-based methods (MISO PD and MISO WTA with $K = 4$ (Sharony et al., 2024), and adapted conditional CPL (Lin et al., 2023)).

NHS achieves recall of 95.78%, 100.00%, and 95.00% on the three tasks, respectively (Tab. 1). While methods such as EPGS and MISO PD are competitive on Ackley, their performance degrades substantially on the other benchmarks, whereas NHS maintains consistently high recall, demonstrating stronger adaptability across diverse nonconvex landscapes.

6.2 CONSTRAINED OPTIMIZATION

We evaluate NHS on nonconvex second-order cone programming (SOCP) and quadratically constrained quadratic programming (QCQP) problems against FSNet (Nguyen & Donti, 2026) and DC3 (Donti et al., 2021). In Tab. 2, NHS achieves zero inequality violation and equality residuals near machine precision on both tasks. Its mean distance to the numerical reference solutions is 15.6% lower than FSNet on SOCP and 31.7% lower on QCQP. These results demonstrate consistently improved solution accuracy together with strong constraint satisfaction across benchmarks.

Table 3: Performance on PGLearn-Small IEEE14 AC-OPF instances.

	BasicNN		PenaltyNN		LDFNN		NHS ($K = 10$)	
Metric ↓	Mean (10^{-3})	Std (10^{-3})	Mean (10^{-3})	Std (10^{-3})	Mean (10^{-3})	Std (10^{-3})	Mean (10^{-3})	Std (10^{-3})
Objective gap (%)	142.237	73.293	765.311	1097.672	151.100	68.878	**0.086**	**0.133**
Active balance violation	12.316	5.266	7.831	0.753	9.152	5.312	$<$ **0.001**	$<$ **0.001**
Reactive balance violation	7.543	1.772	3.088	0.533	7.512	2.398	$<$ **0.001**	$<$ **0.001**
P_g MAE	0.689	0.785	5.172	6.389	0.837	0.679	$<$ **0.001**	**0.001**
Q_g MAE	0.678	0.265	11.111	4.334	0.559	0.235	**0.049**	**0.061**
V_m MAE	0.750	0.322	17.142	6.236	0.647	0.281	**0.006**	**0.004**
V_a MAE (rad)	0.572	0.277	7.607	2.592	0.484	0.270	$<$ **0.001**	$<$ **0.001**
Reference distance (ℓ_2)	5.564	3.020	86.334	17.294	5.181	2.529	**0.146**	**0.185**

We further evaluate NHS on the IEEE14 AC-OPF problem from PGLearn-Small (Klamkin et al., 2025), comparing it with local runs of BasicNN, PenaltyNN, and LDFNN from the official ML4OPF implementation (Klamkin, 2024). As shown in Tab. 3, NHS achieves the lowest mean across all reported metrics, with a mean objective gap of $8.564 \times 10^{-5}\%$ and a mean distance to the reference solutions of 1.464×10^{-4}. These results demonstrate improved solution accuracy and constraint satisfaction, including more accurate reactive power generation predictions and closer agreement with the reference solutions.

6.3 Conditional Probabilistic Sampling

Table 4: Results on sampling problems.

	Mean W_2 ↓		
Method	GMM-2D	DW-1D	DW-4D
ULA	1.5602	0.3150	0.6997
MALA	1.6447	0.3430	0.7047
HMC	1.5448	0.2295	0.7142
SVGD	1.6691	0.9508	1.4500
LIBD	<u>1.2349</u>	<u>0.1849</u>	**0.6219**
DMT	1.9193	1.2935	2.6658
NHS (ours)	**0.6002**	**0.1679**	<u>0.6948</u>

We evaluate NHS on conditional 2D Gaussian mixtures (GMMs) and 1D/4D double-well distributions (Máté & Fleuret, 2023). We compare against ULA (Dalalyan, 2014), MALA (Roberts & Tweedie, 1996), HMC (Neal, 2012), SVGD (Liu & Wang, 2016), LIBD (Máté & Fleuret, 2023) and DMT (Maurais et al., 2025). In Tab. 4, NHS achieves the lowest mean empirical W_2 on GMMs and 1D double wells: 0.6002 and 0.1679, respectively. Although LIBD achieves the lowest error on 4D double wells, NHS remains competitive with a mean W_2 of 0.6948 and maintains consistently strong performance across all three tasks, demonstrating the framework's adaptability to diverse conditional distributions.

6.4 Ablation Study

We examine the predictor and corrector on Rastrigin by comparing the full NHS solver with variants that remove either component. The full solver achieves 100.00% success, compared with 11.72% without the corrector and 5.16% without the predictor (Tab. 12). These deployment ablations show reliance on both jointly trained components. Independent-training controls, homotopy and deployment ablations, and AC-OPF terminal-corrector comparisons are reported in Appendix G.

7 Conclusion

We introduce NHS, a differentiable predictor-corrector framework that replaces the predictor and corrector of a generic homotopy solver with a Neural ODE and a DEQ, coupled through a predictor-anchored proximal term that enables end-to-end training across homotopy levels. Instantiated separately for unconstrained optimization, constrained optimization, and conditional probabilistic sampling, a single trained NHS model amortizes over problem instances within each family, without per-instance retraining, and consistently achieves higher solution accuracy than existing methods across all three settings.

AI USE STATEMENT

In this work, we used generative AI tools to assist with translation, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, and interpreting results. We have not used generative AI tools to generate synthetic datasets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, or implement methods.

We have reviewed all AI-assisted work. All AI-assisted translations were checked against the original text by the authors. AI-assisted data cleaning and reformatting were verified against the original datasets to ensure that no data values or experimental results were unintentionally altered. Qualitative analyses and interpretations suggested with the assistance of generative AI were independently reviewed by the authors and cross-checked against the underlying experimental results before being included in the manuscript. No AI-generated output was incorporated without author review and validation.

We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

David Ackley. *A connectionist machine for genetic hillclimbing*. Springer science & business media, 2012.

Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J Zico Kolter. Differentiable convex optimization layers. *Advances in neural information processing systems*, 32, 2019.

Eugene L Allgower and Kurt Georg. *Numerical continuation methods: an introduction*, volume 13. Springer Science & Business Media, 2012.

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International conference on machine learning*, pp. 136–145. PMLR, 2017.

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

Michael Arbel, Alex Matthews, and Arnaud Doucet. Annealed flow transport monte carlo. In *International Conference on Machine Learning*, pp. 318–330. PMLR, 2021.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.

Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Neural deep equilibrium solvers. In *International Conference on Learning Representations*, 2021.

Daniel J Bates, Andrew J Sommese, Jonathan D Hauenstein, and Charles W Wampler. *Numerically solving polynomial systems with Bertini*. SIAM, 2013.

Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995. doi: 10.1137/0916069.

Omar Chehab, Aapo Hyvarinen, and Andrej Risteski. Provable benefits of annealing for estimating normalizing constants: Importance sampling, noise-contrastive estimation, and beyond. *Advances in Neural Information Processing Systems*, 36:45945–45970, 2023.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Arnak S. Dalalyan. Theoretical guarantees for approximate sampling from smooth and log-concave densities. *arXiv preprint arXiv:1412.7392*, 2014. URL https://arxiv.org/abs/1412.7392.

Priya L Donti, David Rolnick, and J Zico Kolter. Dc3: A learning method for optimization with hard constraints. *arXiv preprint arXiv:2104.12225*, 2021.

Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. Learning iterative reasoning through energy minimization. In *International Conference on Machine Learning*, pp. 5570–5582. PMLR, 2022.

Yilun Du, Jiayuan Mao, and Joshua B Tenenbaum. Learning iterative reasoning through energy diffusion. *arXiv preprint arXiv:2406.11179*, 2024.

Ferdinando Fioretto, Pascal Van Hentenryck, Terrence WK Mak, Cuong Tran, Federico Baldo, and Michele Lombardi. Lagrangian duality for constrained deep learning. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 118–135. Springer, 2020.

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

David M Himmelblau et al. *Applied nonlinear programming*. McGraw-Hill, 2018.

Hidenori Iwakiri, Yuhang Wang, Shinji Ito, and Akiko Takeda. Single loop gaussian homotopy method for non-convex optimization. *Advances in Neural Information Processing Systems*, 35: 7065–7076, 2022.

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

Michael Klamkin. Ml4opf: Machine learning for optimal power flow, 2024.

Michael Klamkin, Mathieu Tanneau, and Pascal Van Hentenryck. PGLearn – an open-source learning toolkit for optimal power flow, 2025. URL https://arxiv.org/abs/2505.22825.

Enming Liang, Minghua Chen, and Steven H Low. Homeomorphic projection to ensure neural-network solution feasibility for constrained optimization. *Journal of Machine Learning Research*, 25(329):1–55, 2024.

Xi Lin, Zhiyuan Yang, Xiaoyuan Zhang, and Qingfu Zhang. Continuation path learning for homotopy optimization. In *International Conference on Machine Learning*, pp. 21288–21311. PMLR, 2023.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. *arXiv preprint arXiv:1608.04471*, 2016. URL https://arxiv.org/abs/1608.04471.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations*, 2019.

Lukas Lüken and Sergio Lucia. Self-supervised learning of iterative solvers for constrained optimization. *Results in Control and Optimization*, pp. 100751, 2026.

Jiayao Mai, Bangyan Liao, Zhenjun Zhao, Yingping Zeng, Haoang Li, Javier Civera, Tailin Wu, Yi Zhou, and Peidong Liu. Neural predictor-corrector: Solving homotopy problems with reinforcement learning. *arXiv preprint arXiv:2602.03086*, 2026.

Vaden Masrani, Rob Brekelmans, Thang Bui, Frank Nielsen, Aram Galstyan, Greg Ver Steeg, and Frank Wood. q-paths: Generalizing the geometric annealing path using power means. In *Uncertainty in Artificial Intelligence*, pp. 1938–1947. PMLR, 2021.

Bálint Máté and François Fleuret. Learning interpolations between boltzmann densities. *arXiv preprint arXiv:2301.07388*, 2023.

Alex Matthews, Michael Arbel, Danilo Jimenez Rezende, and Arnaud Doucet. Continual repeated annealed flow transport monte carlo. In *International Conference on Machine Learning*, pp. 15196–15219. PMLR, 2022.

Aimee Maurais, Bamdad Hosseini, and Youssef Marzouk. Learning paths for dynamic measure transport: A control perspective. *arXiv preprint arXiv:2511.03797*, 2025.

Hossein Mobahi and John W Fisher III. On the link between gaussian homotopy continuation and convex envelopes. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 43–56. Springer, 2015.

Renato DC Monteiro and Ilan Adler. Interior path following primal-dual algorithms. part i: Linear programming. *Mathematical programming*, 44(1):27–41, 1989.

Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

Radford M. Neal. MCMC using hamiltonian dynamics. *arXiv preprint arXiv:1206.1901*, 2012. URL https://arxiv.org/abs/1206.1901.

Hoang Nguyen and Priya Donti. Fsnet: Feasibility-seeking neural network for constrained optimization with guarantees. *Advances in neural information processing systems*, 38:39670–39708, 2026.

Seonho Park and Pascal Van Hentenryck. Self-supervised primal-dual learning for constrained optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4052–4060, 2023.

L. A. Rastrigin. *Extremal Control Systems*. Nauka, Moscow, 1974.

Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

Elad Sharony, Heng Yang, Tong Che, Marco Pavone, Shie Mannor, and Peter Karkus. Learning multiple initial solutions to optimization problems. *arXiv preprint arXiv:2411.02158*, 2024.

Neta Shaul, Juan Perez, Ricky TQ Chen, Ali Thabet, Albert Pumarola, and Yaron Lipman. Bespoke solvers for generative flow models. In *International Conference on Learning Representations*, volume 2024, pp. 12510–12546, 2024.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Chen Xu. Global optimization with a power-transformed objective and gaussian smoothing. *arXiv preprint arXiv:2412.05204*, 2024.

Heng Yang, Pasquale Antonante, Vasileios Tzoumas, and Luca Carlone. Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection. *IEEE Robotics and Automation Letters*, 5(2):1127–1134, 2020.

Guodong Zhang, Kyle Hsu, Jianing Li, Chelsea Finn, and Roger B Grosse. Differentiable annealed importance sampling and the perils of gradient noise. *Advances in Neural Information Processing Systems*, 34:19398–19410, 2021.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36:49842–49869, 2023.

A Full Related Work

Learning for Predictor-Corrector. Predictor-corrector schemes appear in both homotopy continuation and diffusion sampling. Score-based generative models combine numerical prediction with stochastic correction (Song et al., 2020), while UniPC develops a training-free, high-order predictor-corrector solver for pretrained diffusion models (Zhao et al., 2023). Learning can instead target the numerical solution procedure: Bespoke Solvers optimize ODE integrators for pretrained generative flows (Shaul et al., 2024), and neural deep equilibrium solvers learn initializations and fixed-point updates for DEQ inference (Bai et al., 2021). Closer to our setting, NPC formulates homotopy algorithms across several domains as a shared decision process and learns predictor step sizes and corrector termination policies while retaining the underlying numerical state updates (Mai et al., 2026).

Learning for Global Optimization. Classical continuation methods modify the optimization landscape through a family of surrogate problems. Graduated non-convexity progressively introduces nonconvex robust penalties (Yang et al., 2020). Single-loop Gaussian homotopy jointly updates the optimization variable and smoothing scale (Iwakiri et al., 2022). Power-transformed Gaussian smoothing combines objective transformations with smoothing (Xu, 2024). Learning-based approaches target different parts of the optimization process. Learned optimizers amortize iterative updates across problem instances (Andrychowicz et al., 2016), while MISO predicts diverse initial solutions for subsequent optimization (Sharony et al., 2024). Continuation Path Learning represents the solution path with a level-conditioned network and jointly optimizes multiple homotopy subproblems, also connecting this formulation to amortized optimization (Lin et al., 2023). Iterative energy minimization and energy diffusion instead learn task-dependent energy landscapes for reasoning, including annealed energy hierarchies (Du et al., 2022; 2024).

Learning for Constrained Optimization. Learning-based constrained optimization methods differ in how constraints enter training and inference. Lagrangian learning incorporates constraint penalties into model training (Fioretto et al., 2020), while self-supervised primal-dual learning trains primal and dual networks using an augmented-Lagrangian procedure (Park & Van Hentenryck, 2023). OptNet and differentiable convex optimization layers embed parameterized convex programs into neural networks and differentiate through their solutions (Amos & Kolter, 2017; Agrawal et al., 2019). Other methods explicitly repair neural predictions: DC3 combines equality completion with differentiable correction of inequality violations (Donti et al., 2021). FSNet unrolls a feasibility-seeking optimization procedure during training (Nguyen & Donti, 2026). Homeomorphic projection uses an invertible parameterization to enforce feasibility for constraint sets homeomorphic to a unit ball (Liang et al., 2024). LISCO further learns both a primal-dual initializer and an iterative solver using KKT-based self-supervision (Lüken & Lucia, 2026).

Learning for Sampling. Annealed importance sampling bridges tractable and target distributions through intermediate densities, combining Markov transitions with importance weighting (Neal, 2001). Beyond the geometric path, q-paths introduce power-mean interpolations (Masrani et al., 2021), and theoretical analysis establishes the statistical benefits of annealing for normalizing-constant estimation under idealized sampling assumptions (Chehab et al., 2023). Learned approaches adapt either the density path or the transport between its levels. Máté and Fleuret jointly learn endpoint-preserving energy interpolations and transport fields through a continuity-equation objective (Máté & Fleuret, 2023), while Maurais *et al*. formulate path identification as a regularized control problem for measure transport (Maurais et al., 2025). AFT and CRAFT combine learned normalizing-flow transport with sequential Monte Carlo correction, including importance weighting, resampling, and MCMC transitions (Arbel et al., 2021; Matthews et al., 2022). Differentiable AIS enables gradient-based optimization of annealed estimators by removing nondifferentiable Metropolis–Hastings decisions (Zhang et al., 2021).

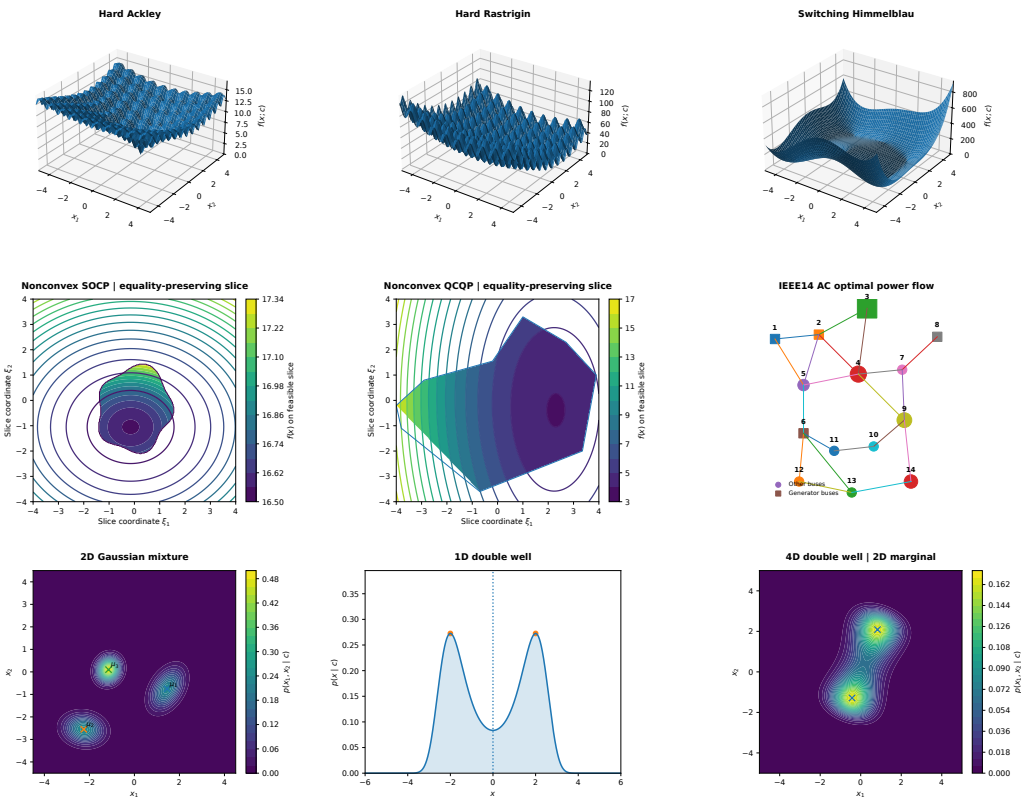

Figure 3: Illustrative views of the nine benchmark problems. Top row, left to right: objective surfaces for Hard Ackley, Hard Rastrigin, and Switching Himmelblau. Middle row: equality-preserving two-dimensional slices of the nonconvex SOCP and QCQP problems, with objective contours and the feasible portion of each slice, followed by the IEEE14 network used for AC optimal power flow (squares denote generator buses). Bottom row: the density of a two-dimensional Gaussian mixture, a one-dimensional double-well density, and a two-dimensional marginal of the four-dimensional double-well distribution. These views illustrate problem geometry.

B Unconstrained Optimization: Experimental Details

B.1 Problem formulation

We minimize $f(x;c)$ over $x \in \mathbb{R}^2$ for $c \sim \mathcal{U}([-1,1]^2)$. Fig. 3 (top row) shows the landscapes. Ackley and Rastrigin use $z = A(c)(x - m(c))$, with

$$\begin{aligned} A(c) &= \begin{pmatrix} \cos\theta/a_1 & \sin\theta/a_1 \\ -\sin\theta/a_2 & \cos\theta/a_2 \end{pmatrix}, \\ \theta &= 0.65\sin(\pi c_1 c_2) + 0.25 c_1, \\ a_1 = \exp(0.35\sin(\pi c_1)), &\qquad a_2 = \exp(0.35\cos(\pi c_2)). \end{aligned} \tag{32}$$

Hard Ackley. The minimizer map and objective are

$$m_A(c) = \begin{pmatrix} 2\sin(\pi c_1/2) + 0.5 c_1 c_2 \\ 1.6\sin(\pi c_2/2) + 0.6(c_1^2 - c_2^2) \end{pmatrix}, \tag{33}$$

$$\begin{aligned} f_A(x;c) = &-20\exp\left(-0.2\sqrt{(z_1^2+z_2^2)/2}\right) \\ &- \exp((\cos(2\pi z_1) + \cos(2\pi z_2))/2) + 20 + e, \end{aligned} \tag{34}$$

Here $z = A(c)(x - m_A(c))$; the reference $m_A(c)$ has objective zero.

Hard Rastrigin.

$$\begin{aligned} m_R(c) &= \begin{pmatrix} 1.8\sin(\pi(c_1 + 0.35c_2)/2) \\ 1.8\sin(\pi(c_2 - 0.35c_1)/2) \end{pmatrix}, \\ \nu(c) &= 0.85 + 0.30\ \mathrm{sigmoid}(2c_1 - c_2), \\ f_R(x;c) &= \sum_{i=1}^{2} \left[z_i^2 + 10\big(1 - \cos(2\pi\nu(c) z_i)\big) \right], \end{aligned} \tag{35}$$

Here $z = A(c)(x - m_R(c))$; the reference $m_R(c)$ has objective zero.

Switching Himmelblau. A context-dependent anchor changes the preferred basin:

$$\begin{aligned} a(c) &= \begin{pmatrix} 3.6\sin(\pi c_1/2) + 0.5 c_1 c_2 \\ 3.6\sin(\pi c_2/2) - 0.5(c_1^2 - c_2^2) \end{pmatrix}, \\ f_H(x;c) &= (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 + \|x - a(c)\|_2^2. \end{aligned} \tag{36}$$

B.2 NHS IMPLEMENTATION

NHS starts at zero and uses ten uniform levels with the field

$$h_\theta(x, s; c) = (1 - s)x + s\nabla_x f(x; c) + s(1 - s) r_\theta(x, s, c). \tag{37}$$

The residual network r_θ is a four-layer, width-128 tanh network. A damped proximal fixed-point corrector follows each prediction. Each task uses 4,096 labeled training contexts, 256 validation contexts. Adam (Kingma & Ba, 2015) takes 1,000 batch-512 updates. The loss combines endpoint supervision, predictor supervision, consistency, and field residuals. Their weights are $(5, 24, 0.025, 0.00025)$ for Ackley and $(5, 16, 0.05, 0.001)$ otherwise.

B.3 BASELINE IMPLEMENTATION

Gaussian-smoothing methods allow 10^4 function evaluations from zero, with queries clipped to $[-6, 6]^2$. GD and L-BFGS-B allow 500 function and gradient evaluations each and retain their native best-query output. MISO uses the official two-layer, width-128 Transformer with four candidates, the same labels, and 1,000 batch-512 updates. CPL uses 1,000 initialization and 1,000 path updates, batch size 512.

B.4 METRICS

Each task uses 128 fresh contexts disjoint from the original 4,480 training, validation, and development-test contexts. We report

$$\begin{aligned} D_f &= \frac{1}{N} \sum_{i=1}^{N} [f(\widehat{x}_i; c_i) - f(x_{\mathrm{ref},i}; c_i)], \\ D_x &= \frac{1}{N} \sum_{i=1}^{N} \|\widehat{x}_i - x_{\mathrm{ref},i}\|_2, \\ \mathrm{Recall}_\epsilon &= \frac{100}{N} \sum_{i=1}^{N} \mathbf{1}\{\|\widehat{x}_i - x_{\mathrm{ref},i}\|_2 \le \epsilon\}. \end{aligned} \tag{38}$$

Recall thresholds are 0.05, 0.01, and 0.1 for Ackley, Rastrigin, and Himmelblau. Results are mean $\pm$ sample SD over five seeds; GD and L-BFGS-B run once. Failures remain in the recall denominator.

Timing. Serial timing uses an RTX 4090, Xeon Platinum 8352V, two CPU threads, and synchronized GPU calls. Times include native correction and candidate selection, but exclude loading, warm-up, references, and metrics.

B.5 Complete results and qualitative examples

Tabs. 5–7 combine accuracy and timing. Figs. 4–6 illustrate the homotopy process and the resulting trajectories; the background contours in the first three panels visualize the squared field-residual norms.

Table 5: Hard Ackley: complete accuracy and inference time.

Method	$D_f \downarrow$	$D_x \downarrow$	Recall (%) $\uparrow$	Time (ms) $\downarrow$
L-BFGS-B	3.0068	1.1335	29.69	2.43401
GD	4.3926	1.6135	7.81	4.22209
GH	0.3420 ± 0.0192	0.0647 ± 0.0021	39.53 ± 5.08	51.99502 ± 0.35297
EPGS	0.1067 ± 0.0069	0.0277 ± 0.0013	92.19 ± 2.07	49.31553 ± 0.17479
PGS	0.2124 ± 0.0243	0.0461 ± 0.0041	71.72 ± 4.15	145.37582 ± 2.63677
ZOSLGH-d	1.0715 ± 0.0464	0.3665 ± 0.0153	56.88 ± 1.50	13.76265 ± 0.51957
ZOSLGH-r	0.2551 ± 0.0330	0.0476 ± 0.0061	72.66 ± 6.08	54.40563 ± 0.78322
MISO PD $K = 4$	0.0887 ± 0.0195	0.0222 ± 0.0042	92.66 ± 2.95	0.11850 ± 0.00123
MISO WTA $K = 4$	0.2277 ± 0.0867	0.0464 ± 0.0144	62.19 ± 21.36	0.11766 ± 0.00080
Conditional CPL	0.1489 ± 0.0559	0.0343 ± 0.0121	80.78 ± 25.03	0.00268 ± 0.00034
NHS (ours)	**0.0649 ± 0.0057**	**0.0175 ± 0.0017**	**95.78 ± 2.38**	6.19626 ± 0.10652

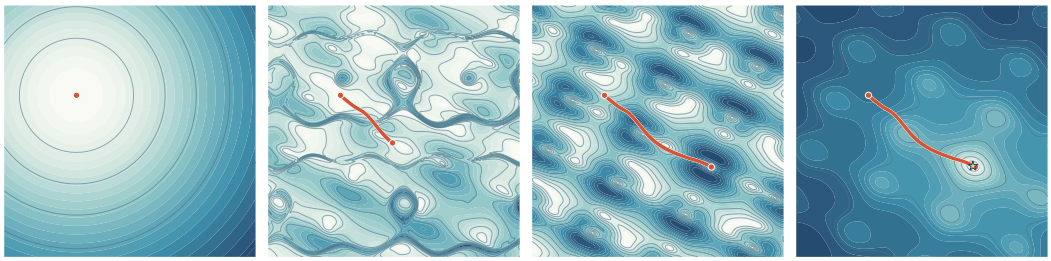

Figure 4: Hard Ackley. From left: squared field residual at $s = 0, 0.5, 1$ with cumulative accepted paths, and target objective with the full path. The fixed-context illustration uses $c = (0.8, -0.6)$; the star denotes the reference solution.

Table 6: Hard Rastrigin: complete accuracy and inference time.

Method	$D_f \downarrow$	$D_x \downarrow$	Recall (%) $\uparrow$	Time (ms) $\downarrow$
L-BFGS-B	5.0549	1.9072	7.03	1.06195
GD	5.1221	1.2958	5.47	24.00838
GH	1.7689 ± 0.1554	0.6261 ± 0.0204	0.94 ± 0.65	47.57807 ± 0.21180
EPGS	4.2337 ± 0.6758	0.2173 ± 0.0237	1.25 ± 0.70	43.25442 ± 0.39985
PGS	4.7798 ± 0.5586	0.1804 ± 0.0163	0.16 ± 0.35	128.63387 ± 1.77765
ZOSLGH-d	7.9029 ± 0.2020	1.7411 ± 0.0020	0.78 ± 0.00	7.18707 ± 0.07121
ZOSLGH-r	1.3524 ± 0.0381	1.0708 ± 0.0145	6.25 ± 1.24	49.49586 ± 1.23446
MISO PD $K = 4$	0.1632 ± 0.1582	0.0208 ± 0.0104	28.59 ± 13.72	0.11348 ± 0.02620
MISO WTA $K = 4$	0.9003 ± 0.4164	0.0531 ± 0.0117	4.38 ± 1.71	0.09787 ± 0.00040
Conditional CPL	1.4851 ± 0.8797	0.5692 ± 0.4777	1.09 ± 1.71	0.00255 ± 0.00005
NHS (ours)	**0.0004 ± 0.0001**	**0.0012 ± 0.0002**	**100.00 ± 0.00**	0.98694 ± 0.01051

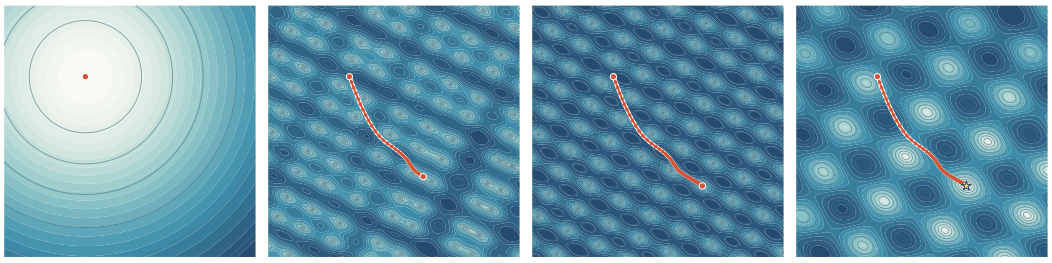

Figure 5: Hard Rastrigin. From left: squared field residual at $s = 0, 0.5, 1$ with cumulative accepted paths, and target objective with the full path. The fixed-context illustration uses $c = (0.8, -0.6)$; the star denotes the reference solution.

Table 7: Switching Himmelblau: complete accuracy and inference time.

Method	$D_f \downarrow$	$D_x \downarrow$	Recall (%) $\uparrow$	Time (ms) $\downarrow$
L-BFGS-B	20.1071	4.1941	30.47	0.70465
GD	20.1071	4.1941	30.47	1.01795
GH	16.3851 ± 0.6717	3.5537 ± 0.0922	27.97 ± 3.51	42.28257 ± 0.23641
EPGS	18.0314 ± 0.4145	3.9024 ± 0.0501	25.78 ± 1.75	39.48272 ± 0.48212
PGS	17.4839 ± 0.5987	3.8034 ± 0.0733	31.09 ± 3.10	123.03666 ± 2.67068
ZOSLGH-d	16.3684 ± 1.0613	3.6927 ± 0.1364	37.97 ± 2.25	11.01069 ± 0.05361
ZOSLGH-r	22.9662 ± 2.5979	4.0931 ± 0.3564	37.19 ± 5.23	43.78988 ± 0.21844
MISO PD $K = 4$	5.0602 ± 1.4327	0.3344 ± 0.0368	32.03 ± 7.61	0.04308 ± 0.00049
MISO WTA $K = 4$	1.5736 ± 2.1368	0.2432 ± 0.1491	50.16 ± 31.19	0.04308 ± 0.00065
Conditional CPL	13.8197 ± 0.1742	3.2330 ± 0.0317	38.59 ± 16.00	0.00250 ± 0.00003
NHS (ours)	**0.0953 ± 0.0827**	**0.0291 ± 0.0076**	**95.00 ± 1.42**	2.34020 ± 0.07696

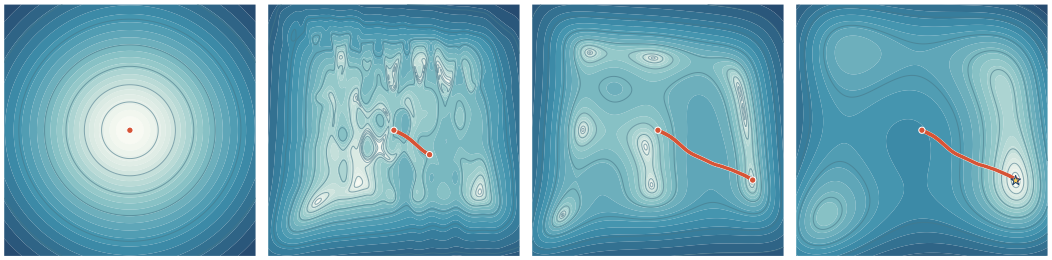

Figure 6: Switching Himmelblau. From left: squared field residual at $s = 0, 0.5, 1$ with cumulative accepted paths, and target objective with the full path. The fixed-context illustration uses $c = (0.8, -0.6)$; the star denotes the reference solution.

C Constrained Optimization: Experimental Details

C.1 Problem formulation

The FSNet families have fixed coefficients, context $b \in \mathbb{R}^{50}$, and decision $x \in \mathbb{R}^{100}$; Fig. 3 (middle row) shows equality-preserving slices. Both minimize

$$f(x) = \tfrac{1}{2} x^\top Q x + p^\top \sin(x), \qquad Ax = b, \qquad -5\mathbf{1} \le x \le 5\mathbf{1}. \tag{39}$$

Nonconvex SOCP. There are 50 additional constraints,

$$\|G_i \cos(x) + h_i\|_2 \le c_i^\top x + d_i, \qquad i = 1, \dots, 50, \tag{40}$$

where $G_i \in \mathbb{R}^{50 \times 100}$, $h_i \in \mathbb{R}^{50}$, $c_i \in \mathbb{R}^{100}$, and $d_i \in \mathbb{R}$.

Nonconvex QCQP. The 50 additional constraints are

$$\tfrac{1}{2}x^\top H_i x + g_i^\top \cos(x) - h_i \le 0, \qquad i = 1, \dots, 50, \tag{41}$$

with $H_i \in \mathbb{R}^{100\times 100}$, $g_i \in \mathbb{R}^{100}$, and $h_i \in \mathbb{R}$. The trigonometric terms make both families nonconvex.

C.2 NHS IMPLEMENTATION

Equality completion sets $x_F = u$ and $x_B = A_B^{-1}(b - A_F u)$. The state contains 50 free primal coordinates, 250 slacks, and 250 multipliers. Ten levels use a conditional Neural ODE predictor and a DEQ-based analytic primal–dual corrector. Both networks have two width-32 tanh hidden layers and 32-dimensional context embeddings. The predictor takes two steps per level. The final float64 solve releases the anchor. Terminal continuation/interior restoration may be used.

The training uses 7,000 cases, with 1,000 validation cases and batch size 512. SOCP uses 1,000 Adam updates at 3×10^{-4}, combining corrected/predicted endpoint MSE (weights 1 and 0.3) and $10f(x)/100$ on numerically admitted rows. QCQP uses AdamW (Loshchilov & Hutter, 2019) at 5×10^{-4}, weight decay 10^{-3}, and equal endpoint MSE weights. It uses 1,192 training updates in total. Final testing uses 2,000 fresh cases per family.

C.3 BASELINE IMPLEMENTATION

FSNet and DC3 replay NHS's batches and cumulative updates: 1,000 for SOCP, 1,192 for QCQP. They retain official losses/correction, with four-layer width-256 MLPs and AdamW.

C.4 METRICS AND RESULTS

We report mean objective difference and full-space L_2 distance to independent local reference solutions. Constraint metrics are the mean and maximum of $\|Ax-b\|_1$ and $\|\max(g(x), 0)\|_1$, including bounds. All outputs enter aggregates; feasibility requires each component's violation to be $\le 10^{-4}$. Infeasible objective values do not establish optimization gains. As shown in Tab. 2, FSNet/NHS are feasible on all 2,000 cases per task. DC3 is feasible on 1,380 SOCP and 194 QCQP cases.

Table 8: Average constrained inference time per instance (ms).

Method	SOCP	QCQP
FSNet	1.22157	0.96713
DC3	0.22655	0.25227
NHS	1395.39983	462.17007

Tab. 8 gives timing on the first 128 fresh cases. Figs. 7 and 8 use qualitative level spacing and fixed primal PCA planes. Intermediate paths join accepted endpoints; only terminal transitions include internal iterates.

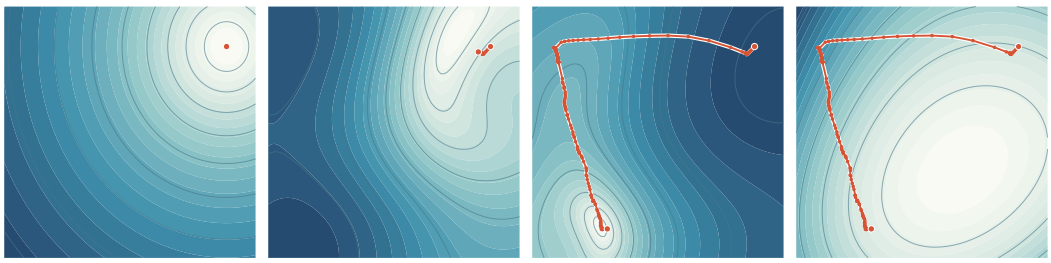

Figure 7: Nonconvex SOCP. From left: squared KKT residual at $s = 0, 0.5, 1$ with cumulative solver paths, and target objective with the full path. The illustrative path is projected onto one fixed primal PCA plane; duals and slacks are held at the corresponding level. Intermediate levels show only the final accepted states; internal iterates are shown only for the terminal solve at $s = 1$.

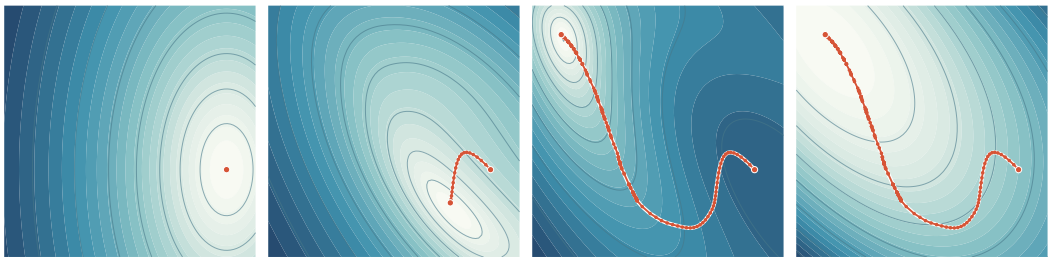

Figure 8: Nonconvex QCQP. From left: squared KKT residual at $s = 0, 0.5, 1$ with cumulative solver paths, and target objective with the full path. The illustrative path is projected onto one fixed primal PCA plane; duals and slacks are held at the corresponding level. Intermediate levels show only the final accepted states; internal iterates are shown only for the terminal solve at $s = 1$.

D AC Optimal Power Flow

D.1 Problem formulation

The PGLearn-Small IEEE14 network has 14 buses, 20 branches, five generators, and 11 loads (Fig. 3, middle right). Context $c = (P_d, Q_d) \in \mathbb{R}^{22}$ specifies loads; $x = (P_g, Q_g, V_m, V_a) \in \mathbb{R}^{38}$ solves

$$\min_x f(x) = \sum_{j=1}^{5} a_j P_{g,j} \quad \text{subject to} \quad h(x; c) = 0, \qquad g(x; c) \le 0. \tag{42}$$

The official ML4OPF equalities impose bus power balance; inequalities impose generation, voltage, angle-difference, and branch thermal limits. Official numerical solutions provide labels and references.

D.2 NHS implementation

Eliminating three fixed active outputs and one angle gauge leaves 34 free primal coordinates, 28 equality multipliers, 122 inequality multipliers, and 122 slacks. Bounded coordinates use affine scaling. The predictor and stationarity correction have two width-eight tanh hidden layers and 32-dimensional context embeddings. Each of the ten levels uses one Euler step of the Neural ODE predictor and a primal–dual Newton solver for the DEQ corrector. The learned correction vanishes at the endpoint.

Training uses 604,964 fitting and 151,241 validation cases, batch size 512, and 1,050 updates. Corrected/predicted endpoint losses have weight one each; consistency has weight 0.01. The terminal forward solve is unanchored. The backward pass uses a primal-anchored shadow equilibrium with adjoint regularization, yielding a surrogate gradient rather than the exact derivative of the forward map.

D.3 Baseline implementation

BasicNN, PenaltyNN, and LDFNN use official ML4OPF code, seed 42, sigmoid widths 500, 300, 500, Adam at 10^{-3}, batch size 32, and 100 epochs (1,890,600 updates). Predictions use the same test cases without repair; the baselines use substantially more optimizer updates than NHS.

D.4 Metrics and results

After aligning voltage angles, we compute

$$G = 100 \frac{|f(\hat{x}) - f(x_{\text{ref}})|}{|f(x_{\text{ref}})|}, \qquad D = \|\hat{x} - x_{\text{ref}}\|_2. \tag{43}$$

Power-balance errors average absolute residuals over buses; variable MAEs average within each physical block. Mean, population SD, and maximum use all 256 cases, including infeasible outputs. Feasibility requires every constraint violation to be $\leq 10^{-4}$; all NHS outputs satisfy it. Tab. 3 reports means/SDs; Tab. 9 gives maxima in original units. Fig. 9 is a qualitative rerun.

Table 9: Per-metric maxima over the same 256 PGLearn instances.

Metric ↓	BasicNN	PenaltyNN	LDFNN	NHS ($K = 10$)
Objective gap (%)	0.46529	6.61093	0.38422	**0.00050**
Active balance violation	0.03082	0.01028	0.02724	$<$ **0.00001**
Reactive balance violation	0.01486	0.00474	0.01585	$<$ **0.00001**
P_g MAE	0.00407	0.04419	0.00389	$<$ **0.00001**
Q_g MAE	0.00250	0.02386	0.00198	**0.00039**
V_m MAE	0.00290	0.02515	0.00243	**0.00002**
V_a MAE (rad)	0.00190	0.01276	0.00144	$<$ **0.00001**
Reference distance (ℓ_2)	0.01706	0.15762	0.01530	**0.00119**

Original, unscaled units with five decimal places; positive values below 0.00001 are shown as $<$ 0.00001. Full-precision Mean, Std and maximum values are preserved in the accompanying CSV.

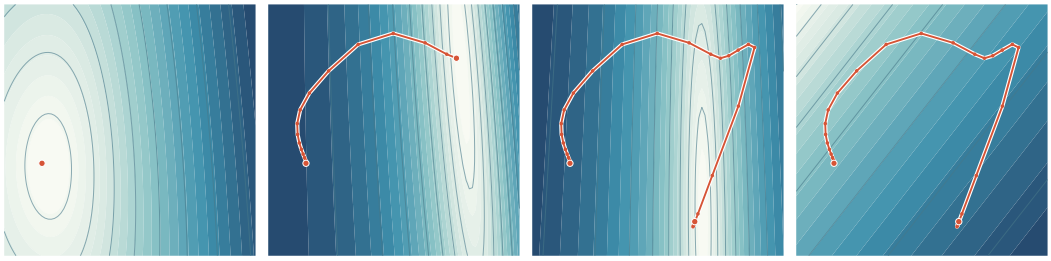

Figure 9: AC optimal power flow. From left: squared KKT residual at $s = 0, 0.5, 1$ with cumulative solver paths, and target objective with the full path. The illustrative path is projected onto one fixed primal PCA plane; duals and slacks are held at the corresponding level. Intermediate levels show only the final accepted states; internal iterates are shown only for the terminal solve at $s = 1$.

E Conditional Sampling

E.1 Problem formulation

The bottom row of Fig. 3 shows the target families. **Gaussian Mixture (2D).**

$$p(x \mid c) = \frac{1}{3}\sum_{k=1}^{3} \mathcal{N}(x; \mu_k, \Sigma_k), \qquad x \in \mathbb{R}^2. \tag{44}$$

Centers are uniform on $[-3, 3]^2$, with pairwise separation ≥ 2. Covariance eigenvalues are log-uniform on $[0.04, 0.25]$ and rotation is uniform on $[0, \pi)$. Context contains the centers and covariance entries; references are drawn directly from the mixture. **Double Well (1D).**

$$p(x \mid m, a, w) \propto \exp[-U_1(x; m, a, w)], \qquad U_1(x; m, a, w) = \frac{((x-m)^2 - a^2)^2}{8a^2w^2}. \tag{45}$$

We sample $m \sim \mathcal{U}[-1.5, 1.5]$, $a \sim \mathcal{U}[1.5, 2.5]$, and $w \sim \mathcal{U}[0.5, 0.8]$, with context $(m, 1, a, w)$. **Double Well (4D).** For unit v, set $z = v^\top(x - m)$ and $r = x - m - zv$:

$$p(x \mid m, v, a, w) \propto \exp[-U_4(x; m, v, a, w)], \qquad U_4 = \frac{(z^2 - a^2)^2}{8a^2w^2} + \frac{\|r\|^2}{2w^2}. \tag{46}$$

Each center coordinate is uniform on $[-1.5, 1.5]$; v is uniform on the unit sphere, with the same a, w distributions. Both double wells have two equal-mass modes. Axial references use numerical

inverse-CDF sampling on 8,193 points over $[-3, 3]$ in units of a; 4D references add orthogonal Gaussian noise of scale w.

E.2 NHS IMPLEMENTATION

Particles start from $\mathcal{N}(0, I_d)$ with energy

$$H_\theta(x, s, c) = (1 - s)U_0(x) + sU_c(x) + s(1 - s)b_\theta(x, s, c), \qquad b_\theta = 5\tanh(f_\theta/5), \tag{47}$$

Both networks have four width-64 tanh hidden layers. Each of ten levels uses two Euler predictor steps and five deterministic Stein corrector steps, step size 0.05 and RBF bandwidth 0.5. A Wasserstein proximal term of weight one anchors particles to the predicted cloud by optimal assignment.

Seeds each use 2,000 Adam updates. Batches contain 256 fresh contexts with 256 initial/reference particles each. The loss combines terminal squared W_2 for the full and predictor-only rollouts (weight one each) and consistency (weight 0.1). Rates decay from 10^{-3} to 10^{-4} over 2,000 updates for double wells. GMM decays over 1,000 then stays fixed. Checkpoint selection minimizes W_2 on 32 validation contexts, every 100 updates with 400 particles. GMM selection uses the second 1,000-update phase.

E.3 BASELINE IMPLEMENTATION

ULA, MALA, HMC, and SVGD use 200 score evaluations per particle and validation-selected steps; HMC uses ten leapfrog steps per proposal. LIBD uses the authors' implementation, batch size 64, and 5,001 updates per target. Our DMT adaptation uses nine temporal nodes and at most 60 solver function evaluations, with spatial grids of 17×17 (GMM), 50 points (DW-1D), or 128 Sobol points (DW-4D). Both generate with 100 Euler steps and reuse each target fit across three generation seeds.

E.4 METRICS AND RESULTS

Each task uses 128 held-out targets and 400 particles. All methods share initial and reference clouds.

$$\widehat{W}_2(X, Y) = \left[\frac{1}{N}\min_{\pi \in S_N}\sum_{i=1}^{N}\|x_i - y_{\pi(i)}\|^2\right]^{1/2}. \tag{48}$$

Eq. 48 uses sorted matching in 1D and optimal assignment in 2D/4D. Each model averages 384 distances. NHS mean and sample SD use four training seeds. LIBD/DMT fitting-seed variability is not assessed. Tab. 4 compares methods. Tab. 10 gives per-seed scores. Figs. 10–12 show particle evolution. DW-4D metrics use the full state, not its projection.

Table 10: Per-training-seed mean W_2 (↓) and validation-selected optimizer update (in parentheses). Each mean averages 128 targets and three generation seeds.

Training seed	GMM (2D) ↓	DW (1D) ↓	DW (4D) ↓
2	0.6023 (1,900)	<u>0.1674</u> (1,800)	0.6970 (1,900)
3	<u>0.5972</u> (1,200)	0.1689 (1,500)	0.6969 (1,700)
5	0.6090 (1,600)	0.1690 (1,800)	<u>0.6927</u> (2,000)
7	**0.5924** (2,000)	**0.1664** (1,800)	**0.6925** (2,000)

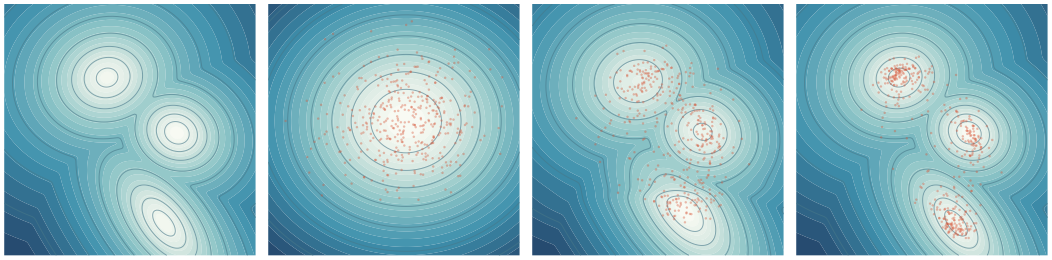

Figure 10: Two-dimensional Gaussian mixture. From left: target energy and learned homotopy energies at $s = 0, 0.5, 1$. Red points show the 400-particle states at each level.

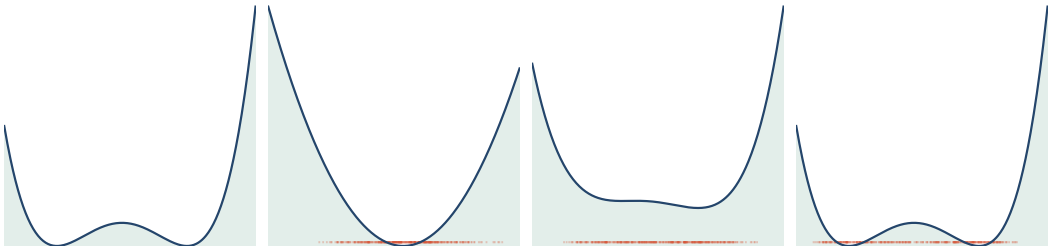

Figure 11: One-dimensional double well. From left: target energy and learned homotopy energies at $s = 0, 0.5, 1$. Red points show the 400-particle states at each level.

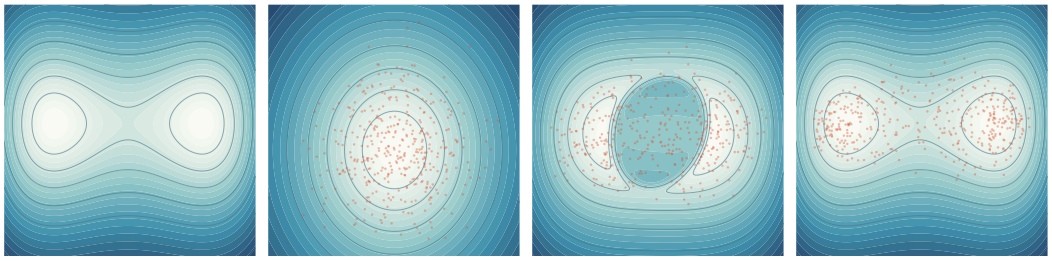

Figure 12: Four-dimensional double well. From left: target energy and learned homotopy energies at $s = 0, 0.5, 1$. Red points show the 400-particle states at each level. Energy maps are fixed two-dimensional slices; particle locations are projections of the four-dimensional states.

F Qualitative Analysis of Predictor–Corrector Training

Fig. 13 uses optimization objective $f(x; c) = (x^2 - 1)^2 + cx$ at $c = 0.8$, easy objective $g(x) = \frac{1}{2}(x - 1)^2$, and $H_\theta = (1 - s)g + sf + s(1 - s)V_\theta$. This is separate from double-well sampling. Each epoch is one update on 256 contexts; the predictor learns from consistency, while the corrector also receives endpoint supervision. **Epoch 0.** The linear homotopy stays in the inferior right well, with objective gap 1.62. **Epoch 100.** The learned path redirects toward the left well; the gap falls to 3.74×10^{-3}. **Epoch 800.** The gap reaches 7.57×10^{-6} and predictor–corrector RMSE 4.99×10^{-4}, versus 3.18×10^{-3} initially.

G Additional Ablation Results

G.1 Frozen-checkpoint deployment ablations on Hard Rastrigin

Shared Setting and Evaluation. We evaluate five existing Hard Rastrigin checkpoints, trained with seeds $2, 3, 5, 7, 11$, on the same 128 fresh contexts used in the main comparison. The context-dependent landscape is illustrated in Figure 5. The problem definition and data protocol are given in the preceding experimental sections. All weights and deployment choices are frozen before fresh testing. The reference configuration uses Linear homotopy, both predictor and corrector, $L = 10$ intervals, predictor scale $\alpha = 1$, and a per-level corrector cap $K = 5$, with weights trained using $K = 20$. This configuration was selected during earlier development. The results below are not used to select a new configuration.

Each deployment ablation changes one setting while keeping the remaining reference settings fixed. Gap denotes the mean objective difference from the reference solution, and L2 denotes the mean Euclidean distance to that solution. Tables report both in units of 10^{-3}. SR is the percentage of contexts with L2 distance at most 0.01. We first aggregate over contexts within each run and then report the mean and sample standard deviation across the five checkpoints. The retained ablation evaluator uses float32, whereas the main-table target evaluation uses float64, so small differences in reported objective gaps can occur for the same deployment.

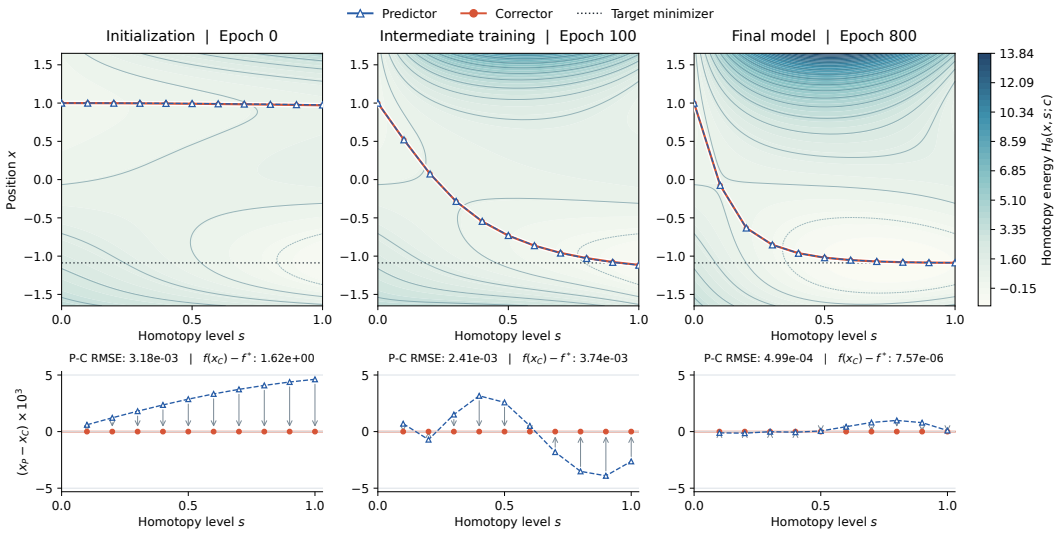

Figure 13: Training evolution for unconstrained 1D double-well optimization at $c = 0.8$. Columns show epochs 0, 100, and 800. Top: the scalar homotopy energy on a shared color scale, with predictor states (blue triangles), corrected states (red circles), and the target minimizer (dotted line). Bottom: the signed predictor displacement relative to its corrected state, $(x_P - x_C) \times 10^3$, on a shared scale. Arrows indicate the correction. All states are recorded solver outputs.

Homotopy Form. We replace the Linear anchor with the Geometric alternative at inference, retaining the learned parameters and the same predictor–corrector settings. Table 11 shows that the mean gap increases from 0.368×10^{-3} to 7.873×10^{-3}, while mean L2 increases from 1.196×10^{-3} to 2.648×10^{-3}. SR decreases from 100.00% to 96.72%, with a larger gap standard deviation under the replacement. Thus, the fitted solver is sensitive to changing its path at deployment. Since both rows use Linear-trained weights, this experiment does not rank independently trained homotopy families. Geometric homotopy nevertheless remains a viable alternative and an interesting direction for future investigation.

Table 11: Homotopy form on the fresh Rastrigin holdout.

Form	Gap (10^{-3}) ↓	L2 (10^{-3}) ↓	SR (%) ↑
Linear	**0.368 ± 0.133**	**1.196 ± 0.188**	**100.00 ± 0.00**
Geometric	7.873 ± 12.185	2.648 ± 1.974	96.72 ± 4.47

Mean ± s.d. across five fixed checkpoints.

Predictor and Corrector. We disable correction for P only and replace prediction by an identity update for C only, retaining the jointly trained weights. Table 12 reports SR of 11.72% for P only and 5.16% for C only, compared with 100.00% for P+C. Their respective mean L2 distances are 55.492×10^{-3}, 1648.192×10^{-3}, and 1.196×10^{-3}. Both components therefore contribute to the accuracy of this fitted solver: neither removal preserves its performance.

Table 12: Predictor and corrector on the fresh Rastrigin holdout.

Variant	Gap (10^{-3}) ↓	L2 (10^{-3}) ↓	SR (%) ↑
P only (joint)	1298.171 ± 1118.108	55.492 ± 25.576	11.72 ± 4.42
C only (joint)	3850.949 ± 171.435	1648.192 ± 37.466	5.16 ± 0.43
P+C	**0.368 ± 0.133**	**1.196 ± 0.188**	**100.00 ± 0.00**

Mean ± s.d. across five fixed checkpoints.

Number of Homotopy Intervals. We use a linear time grid with $L \in \{1, 2, 4, 8, 10, 16, 32\}$ while fixing $K = 5$ at each level. The learned weights are unchanged. In Table 13, $L = 1$ and $L = 2$ yield zero SR, and $L = 4$ reaches only 21.25%. Increasing to $L = 8$ raises SR to 97.50%; the reference $L = 10$ achieves 100.00% and the lowest mean gap and L2 in this sweep. Further increasing to $L = 16$ or 32 lowers SR to 96.56% or 92.97%. A finer grid therefore does not monotonically improve this frozen model. Changing L also changes the total correction budget, so this is a discretization-sensitivity study, not a matched-cost comparison.

Table 13: Number of homotopy intervals on the fresh Rastrigin holdout.

L	Gap (10^{-3}) ↓	L2 (10^{-3}) ↓	SR (%) ↑
1	43475.732 ± 1947.319	4350.844 ± 141.322	0.00 ± 0.00
2	19897.686 ± 2369.342	1732.678 ± 234.086	0.00 ± 0.00
4	6285.312 ± 602.528	233.678 ± 89.004	21.25 ± 6.71
8	19.626 ± 25.394	5.218 ± 5.016	97.50 ± 2.10
10	**0.368 ± 0.133**	**1.196 ± 0.188**	**100.00 ± 0.00**
16	118.478 ± 78.375	18.279 ± 6.577	96.56 ± 1.42
32	340.186 ± 315.168	45.560 ± 7.661	92.97 ± 1.99

Mean ± s.d. across five fixed checkpoints.

Predictor Update Scale. We multiply the predictor increment by $\alpha \in \{0, 0.25, 0.5, 1, 1.5, 2\}$ without changing the time grid or the number of integration substeps. Table 14 shows that suppressing prediction ($\alpha = 0$) reproduces C only, with SR 5.16%. SR increases to 17.66% and 72.66% at scales 0.25 and 0.5, and reaches 100.00% at the reference scale 1. Larger updates also degrade accuracy: scales 1.5 and 2 give 96.88% and 78.75% SR. The result supports retaining the update scale used by the fitted predictor. Both attenuating and amplifying its increments can impair tracking under fixed correction settings.

Table 14: Predictor update scale on the fresh Rastrigin holdout.

α	Gap (10^{-3}) ↓	L2 (10^{-3}) ↓	SR (%) ↑
0	3850.949 ± 171.435	1648.192 ± 37.466	5.16 ± 0.43
0.25	1886.928 ± 274.248	818.854 ± 26.340	17.66 ± 1.52
0.5	382.523 ± 119.700	130.958 ± 14.701	72.66 ± 3.66
1	**0.368 ± 0.133**	**1.196 ± 0.188**	**100.00 ± 0.00**
1.5	14.779 ± 9.156	9.434 ± 2.346	96.88 ± 1.66
2	30.155 ± 11.750	15.193 ± 9.472	78.75 ± 10.28

Mean ± s.d. across five fixed checkpoints.

Corrector Iteration Cap. We vary the maximum inner proximal fixed-point iterations per level, $K \in \{0, 1, 2, 5, 10, 20\}$, retaining weights trained with $K = 20$ and the original batch-wide early-stopping rule. In Table 15, $K = 0$ reproduces P only. One and two iterations raise SR to 92.97% and 97.19%; the reference cap $K = 5$ reaches 100.00%. Caps 10 and 20 both give 99.84%, without reducing mean gap or L2 below the $K = 5$ values. More allowed correction therefore does not ensure better endpoint accuracy for these fixed weights. The $K = 20$ row preserves the earlier deployment for comparison; the cap is a maximum, not a measurement of realized work.

Table 15: Corrector iteration cap on the fresh Rastrigin holdout.

K	Gap (10^{-3}) ↓	L2 (10^{-3}) ↓	SR (%) ↑
0	1298.171 ± 1118.108	55.492 ± 25.576	11.72 ± 4.42
1	170.103 ± 369.597	6.446 ± 7.754	92.97 ± 7.07
2	149.664 ± 205.652	5.368 ± 4.827	97.19 ± 4.64
5	**0.368 ± 0.133**	**1.196 ± 0.188**	**100.00 ± 0.00**
10	0.573 ± 0.395	1.239 ± 0.194	99.84 ± 0.35
20	0.715 ± 0.761	1.229 ± 0.196	99.84 ± 0.35

Mean ± s.d. across five fixed checkpoints.

Training versus Deployment. The correction budget used while fitting the predictor is distinct from the budget used at inference. On the same fresh holdout, the retained $K = 20$-trained/$K = 5$-deployed models achieve 100.00% SR, whereas the existing independently $K = 5$-trained/$K = 5$-

deployed models achieve 95.62%. Reducing the inference cap of an existing model is therefore not equivalent to training with that smaller cap.

G.2 AC-OPF TERMINAL-CORRECTOR ABLATION

We test whether direct terminal correction of a learned primal prediction can reproduce the accuracy and feasibility of the complete NHS path on IEEE14 (an objective slice and solution trajectory are shown in Figure 9). The two additional BasicNN + TC baselines use the identical saved BasicNN predictions on indices 512–767. Physical predictions are gauge-aligned and inverse-affine mapped to the existing reduced primal coordinates; fixed coordinates take their prescribed values. The dual and slack variables are initialized using the standard NHS initialization, while only the primal variables are replaced by the BasicNN prediction.

Table 16: Terminal-corrector ablation on the same 256 IEEE14 instances as Table 3.

Metric	BasicNN	TC10	TC40	NHS
Obj. gap (%) ↓	0.14224 ± 0.07329	4.63300 ± 13.09794	3.92752 ± 12.80076	**0.00009 ± 0.00013**
Active bal. ↓	0.01232 ± 0.00527	0.02557 ± 0.08376	0.02467 ± 0.08347	**< 0.00001 ± < 0.00001**
Reactive bal. ↓	0.00754 ± 0.00177	0.00856 ± 0.02957	0.00828 ± 0.02952	**< 0.00001 ± < 0.00001**
P_g MAE ↓	0.00069 ± 0.00078	0.04755 ± 0.13673	0.04143 ± 0.13442	**< 0.00001 ± < 0.00001**
Q_g MAE ↓	0.00068 ± 0.00027	0.05346 ± 0.08744	0.02814 ± 0.09353	**0.00005 ± 0.00006**
V_m MAE ↓	0.00075 ± 0.00032	0.04176 ± 0.12574	0.03686 ± 0.12660	**< 0.00001 ± < 0.00001**
V_a MAE ↓	0.00057 ± 0.00028	0.02660 ± 0.08743	0.02549 ± 0.08642	**< 0.00001 ± < 0.00001**
Ref. L2 ↓	0.00556 ± 0.00302	0.33420 ± 0.77151	0.23934 ± 0.78830	**0.00015 ± 0.00018**
Feasible (%) ↑	0.00	22.27	89.84	**100.00**

TC10/TC40: BasicNN plus terminal correction with $K = 10/40$. Mean ± population s.d. includes all 256 outputs. Balance entries are violations; V_a MAE is in radians. Positive values below 10^{-5} are shown as < 0.00001. Feasibility requires every official violation to be at most 10^{-4}.

Each baseline makes one direct call to the same unanchored terminal Newton solver at $s = 1$, with $\mu_{\text{end}} = 10^{-7}$, float64 arithmetic, the same residual equations, positivity control, Armijo line search, and KKT stopping tolerance 10^{-6}. The learned predictor and all intermediate homotopy levels are skipped. No recovery, additional polishing, or parameter updates are allowed. The sole numerical setting changed between these baselines is the iteration cap, $K = 10$ versus $K = 40$. Both runs start from bitwise-identical initial states, and their first ten Newton traces match exactly.

The $K = 10$ run uses all ten iterations on every instance and reaches the physical feasibility threshold on $57/256$ instances, although none satisfies the stricter KKT stopping criterion. At $K = 40$, $230/256$ satisfy both criteria; 23 fail to converge at the cap and three stop earlier without accepting a new Newton step. One successful instance converges on iteration 40. All finite outputs, including failures, enter Tab. 16.

Accuracy and Feasibility. Increasing the terminal cap from 10 to 40 raises feasibility from $57/256$ (22.27%) to $230/256$ (89.84%), compared with $256/256$ for NHS. The all-instance mean objective gaps are 4.633%, 3.928%, and $8.564 \times 10^{-5}\%$, respectively. Unsuccessful outputs are not excluded; their objective gaps do not certify feasible optimality. The $K = 40$ baseline averages 18.30 iterations because of early stopping, whereas NHS averages 40.72 over its full path. The cap therefore provides an approximate iteration-budget comparison, not matched realized work. Under the stated initialization, direct terminal correction improves feasibility but does not reproduce NHS's overall accuracy and reliability. These results demonstrate the value of the full NHS homotopy procedure, which achieves higher solution accuracy and more reliable feasibility than direct terminal correction.

Table 17: Iteration counts and amortized inference costs.

Method	Mean steps	ms/instance
BasicNN	0	0.00641
BasicNN + TC10	10.000	47.687
BasicNN + TC40	18.305	231.933
NHS	40.723	187.011

BasicNN: historical forward time (batch 8192). TC: estimated totals; correction-only times are 47.680/231.927 ms. Correction timing uses synchronized GPU wall time at batch 128, including initialization, correction, trace formatting, and decoding; model/data loading and metric evaluation are excluded. NHS: earlier same-server RTX 4090 rerun. Counts include convergence-check iterations and all 10 NHS levels.

Realized Work and Timing. Table 17 separates iteration counts from amortized

runtime. The $K = 40$ terminal baseline averages 18.305 Newton iterations and an estimated 231.933 ms per instance; NHS averages 40.723 iterations over all 10 levels and 187.011 ms. An iteration cap therefore does not match realized work or runtime. Terminal correction uses float64 throughout; NHS uses float32 at intermediate levels and float64 at its endpoint. Line-search costs and active batch sizes also differ.

The same-server NHS rerun remains feasible on all 256 instances. The accuracy comparison retains the original Table 3 metrics. The TC totals combine measured correction with historical BasicNN forward time, and are estimates rather than matched end-to-end measurements.