PACE: PARAMETER-EFFICIENT CALIBRATION FOR ROBUST POSTERIOR INFERENCE UNDER SIMULATOR MISSPECIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Simulator based inference provides a framework for inferring parameters of complex stochastic simulators in the absence of explicit likelihood. However, misspecification in the simulator often creates systemic discrepancies, which leads to poor posterior estimates for real data samples. Recent works have attempted to address this by explicitly handling the discrepancy using methods such as transport-based correction, or robust summary learning. However, this involves additional objectives or multi-stage procedures. We show that, surprisingly, such elaborate adaptation is often not required in many cases of practical interest. We introduce PACE, a simple and deliberately minimal calibration strategy for simulator-pretrained neural posterior inference. Rather than defining explicit correction mechanism, PACE levareges the representation and posterior structure learnt using abundant simulated data and calibrates the posterior using light weight feature wise transformations learnt using a small set of labeled real observations. These light-weight transformations are parameterized using as a small set of calibration parameters and optimized directly for posterior inference. Despite its compact calibration space and simple strategy, PACE demonstrates effective outcome across diverse data modalities ranging from timeseries to images. On multiple evaluation metrics, PACE yields comparable or better results than SOTA approaches while requiring adaptation of a very small number of parameters. Our results suggest that when simulation-trained representations retain the information required for posterior inference, expensive re-learning or explicit alignment to the target distribution is unnecessary and a lightweight calibration of existing inference pathway suffices.

1 INTRODUCTION

Across many areas of science and engineering, simulators are used to model complex systems and understand how their underlying parameters generate the observations. However, these parameters are often difficult to measure directly or to determine experimentally. Reliable and uncertainty-quantified estimation of these parameters from observations remains an important problem for understanding these systems. In many real-world settings, however, the likelihood of the observed data under the simulator is either unavailable or computationally intractable. Simulation-based inference (SBI) provides a framework for Bayesian parameter inference when the likelihood is unavailable or intractable by learning directly from data generated by the simulator (Cranmer et al., 2020). Modern SBI methods use neural networks to learn surrogate models of the likelihood (Papamakarios et al., 2019), likelihood-to-evidence ratio (Hermans et al., 2020), or posterior distribution (Papamakarios & Murray, 2016). In particular, neural posterior estimation (NPE) learns a conditional density estimator $q_\phi(\theta \mid x)$ that directly approximates the posterior over parameters given an observation. Density estimators, such as normalizing flows, are appealing and they allow NPE to capture complex and potentially non-Gaussian posteriors. An especially important advantage of NPE is *amortization*: once trained, it can quickly estimate the posterior for new data without retraining or additional simulator evaluations. This makes NPE particularly attractive when inference must be performed repeatedly for new observations.

However, the success of NPE depends critically on how well the simulator represents the actual data-generating process. In practice settings, simulators are approximations of complex systems and may

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.

omit relevant mechanisms, rely on simplifying assumptions, or fail to capture important characteristics of observed data. Such mismatch between simulated and observed data, commonly referred to as *simulator misspecification*, can lead to inaccurate or poorly calibrated posterior estimates (Cannon et al., 2022; Schmitt et al., 2023). Addressing simulator misspecification while preserving fast amortized inference is therefore an important challenge in SBI: ideally, correcting the posterior should not require expensive optimization or additional simulator evaluations for every new observation.

Recently, several approaches have been proposed to address simulator misspecification in neural posterior estimation (NPE). Most of these proposed approaches treats simulator to reality gap as distribution alignment problem. Robust Neural Posterior Estimation (RNPE) explicitly models discrepancies between simulated and observed data (Ward et al., 2022), while other approaches seek summary representations that are less sensitive to misspecification (Huang et al., 2023). More recently, RoPE proposed leveraging a small labeled calibration set and used optimal transport to establish correspondences between simulated and observed representations (Wehenkel et al., 2025). ROPE's inference step however is transductive, requiring access to test observations during the correction stage. Subsequently, FRISBI addressed this issue by learning an inductive correction that enables amortized inference on unseen observations (Senouf et al., 2025). However, FRISBI uses a multi-stage procedure which first performs supervised and optimal-transport-based alignment, and amortizes the resulting correction using a conditional normalizing flow.

This progressions raise a natural question: *How much of simulator trained inference need to change to handle simulator to reality gap?* The answer depends on how fundamentally misspecification affects the pretrained estimator. In some situations where the simulator substantially differs from the real process, the representations and posterior structure learned from simulated data may not be transferrable to real observations. Fixing this may therefore require substantial model adaptation or aligning the simulated and observed data explicitly. In contrast, in many other situations, much of what the pretrained model has learned from simulations may still be transferrable to real observations, with the correction of misspecification needed only a small adaptation of the learned representations.

Our work considers the second hypothesis with PACE, in which we deliberately perform only a restricted calibration of simulator pretrained NPE. Once pretrained on abundant simulated data, the entire simulator trained inference model is held frozen. During the correction step, given only a small set of real observations paired with their known parameters, we learn feature-wise transformations of the learned representations along the existing inference pathway. Crucially, no simulator-trained transformation is updated. Neither does the method require the discrepancy distribution to be specified nor do the simulated and observed distributions have to have to explicitly matched via distribution alignment or optimal transport. The correction step is simply a parameter-efficient calibration performed directly through the posterior objective by minimizing the negative log posterior probability of the known target parameters. The simplicity of PACE is intentional. Our goal is not to construct a more expressive domain-adaptation model, but to understand how much adaptation is actually required to correct simulator misspecification, especially when only limited calibration data are available. Our main contributions are:

- **A different hypothesis about misspecified SBI.** We investigate how much adaptation is actually required to bridge the simulated-to-real gap, and whether explicit discrepancy modeling or distribution alignment is necessary to solve the misspecified SBI problem.
- **Parameter efficient calibration for misspecified SBI.** Our method PACE introduces a lightweight calibration space based on feature-wise scale-and-shift transformations while keeping the pretrained NPE frozen. The parameters defining the calibration are learned using a small labeled calibration set using the posterior objective itself. Notably, PACE requires neither an explicit alignment objective nor optimal transport and remains fully amortized after calibration.
- **Strong performance across diverse data modalities and benchmarks.** We evaluate PACE across various benchmarks covering different observation types and both synthetic and real-world settings. It achieves performance comparable to or better than state-of-the-art approaches across multiple evaluation metrics, while introducing only about 0.5–4% additional trainable parameters.
- **Fast amortized inference on unseen observations.** PACE retains the key advantage of amortized SBI. Once calibrated, the same estimator can be applied independently to new observations without test-time optimization, additional simulator evaluations, or access to other test observations.

2 Background and Problem Formulation

Simulator setup. Computer simulators are widely used to model complex real-world processes and phenomena for which direct experimentation may be difficult, costly, or impractical. A simulator provides a computational model to understand how underlying parameters give rise to observable data. We consider a simulator $\mathcal{S}$ that generates observations from a set of underlying parameters. Let $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^k$ denote the parameters of interest, and let $\boldsymbol{x}_s \in \mathcal{X}_s \subseteq \mathbb{R}^d$ and $\boldsymbol{x}_o \in \mathcal{X}_o \subseteq \mathbb{R}^d$ denote simulated and real observations, respectively. For a given $\boldsymbol{\theta}$, the simulator defines a conditional distribution $p_s(\boldsymbol{x} \mid \boldsymbol{\theta})$ from which simulated observations $\boldsymbol{x}_s$ can be generated. We assume a prior distribution $p(\boldsymbol{\theta})$ over the parameters, which approximates the marginal distribution $p^\star(\boldsymbol{\theta})$ of parameter values in the real world. Together, $p(\boldsymbol{\theta})$ and $p_s(\boldsymbol{x} \mid \boldsymbol{\theta})$ induce the joint distribution $p_s(\boldsymbol{\theta}, \boldsymbol{x}_s) = p(\boldsymbol{\theta})p_s(\boldsymbol{x}_s \mid \boldsymbol{\theta})$ over simulated parameter–observation pairs $(\boldsymbol{\theta}, \boldsymbol{x}_s)$. In many scientific applications, although the simulator can generate observations for different parameter values, evaluating the corresponding likelihood $p_s(\boldsymbol{x} \mid \boldsymbol{\theta})$ is analytically intractable or computationally prohibitive. The objective is therefore to infer the posterior distribution over the underlying parameters $\boldsymbol{\theta}$ from an observation $\boldsymbol{x}$ without requiring explicit evaluation of the likelihood.

Simulation-based inference. Simulation-based inference (SBI) provides a framework for Bayesian parameter inference when the likelihood is unavailable or intractable, but samples can be generated from a simulator (Cranmer et al., 2020; Deistler et al., 2025). Modern SBI methods use machine learning to learn approximations required for inference directly from simulator-generated samples. For example, neural likelihood estimation learns an approximation of the likelihood (Papamakarios et al., 2019), while neural ratio estimation learns likelihood ratios using classification-based objectives (Hermans et al., 2020). In this work, we focus on neural posterior estimation (NPE) (Papamakarios & Murray, 2016), which directly learns an approximation of the posterior distribution over the parameters conditioned on an observation. For an observation $\boldsymbol{x}$, the posterior $p_s(\boldsymbol{\theta} \mid \boldsymbol{x})$ describes the plausibility of different parameter values under the simulator after observing $\boldsymbol{x}$. NPE approximates this posterior using a neural conditional density estimator $q_\phi(\boldsymbol{\theta} \mid \boldsymbol{x})$ trained on simulated parameter–observation pairs, such that $q_\phi(\boldsymbol{\theta} \mid \boldsymbol{x_s}) \approx p_s(\boldsymbol{\theta} \mid \boldsymbol{x_s})$.

For high-dimensional or structured observations, NPE commonly employs a *neural summary encoder* (NSE) $\boldsymbol{h_\psi} : \mathcal{X} \to \mathbb{R}^m$ to map an observation $\boldsymbol{x}$ to a lower-dimensional representation $\boldsymbol{z} = \boldsymbol{h_\psi}(\boldsymbol{x})$. The representation $\boldsymbol{z}$ is then used as conditioning information for a conditional density estimator (CDE), which models the distribution over $\boldsymbol{\theta}$ conditioned on $\boldsymbol{z}$. In this work, we consider normalizing-flow-based CDEs (Papamakarios et al., 2017; Durkan et al., 2019; Papamakarios et al., 2021), which construct flexible conditional distributions through a sequence of invertible transformations. Combining the NSE and CDE, the complete NPE can be written as $q_{\boldsymbol{\phi},\boldsymbol{\psi}}(\boldsymbol{\theta} \mid \boldsymbol{x_s}) = q_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \boldsymbol{h_\psi}(\boldsymbol{x_s}))$.

The NSE and CDE are trained jointly on simulated parameter–observation pairs by minimizing the negative conditional log-likelihood

$$\mathcal{L}_{\mathrm{NPE}}(\boldsymbol{\phi}, \boldsymbol{\psi}) = -\mathbb{E}_{(\boldsymbol{\theta},\boldsymbol{x}_s)\sim p_s(\boldsymbol{\theta},\boldsymbol{x}_s)} \left[\log q_{\boldsymbol{\phi},\boldsymbol{\psi}}(\boldsymbol{\theta} \mid \boldsymbol{x}_s)\right]. \tag{1}$$

Importantly, this objective can be optimized using samples generated by the simulator without evaluating the simulator likelihood. The NSE learns representations useful for posterior inference, while the CDE maps these representations to distributions over the underlying parameters. Once trained, inference is *amortized*: a new observation x can be passed through the same trained NPE to directly obtain an approximate posterior over θ, without retraining the model or performing additional simulator evaluations.

Simulator misspecification. The NPE is trained on observations generated by the simulator, and its application to real observations therefore relies on the simulator adequately representing the underlying data-generating process. In practice, simulators are necessarily approximations of real systems and may simplify or omit relevant mechanisms, nuisance effects, or characteristics of the observation process. Consequently, simulated and real observations may systematically differ even when they correspond to the same underlying parameter value. We denote the conditional distribution of real observations by $p_o(\boldsymbol{x} \mid \boldsymbol{\theta})$, such that $x_o \sim p_o(\boldsymbol{x} \mid \boldsymbol{\theta})$. Simulator misspecification arises when the simulated and real data-generating processes differ, i.e., $p_s(\boldsymbol{x} \mid \boldsymbol{\theta}) \neq p_o(x \mid \theta)$. As a result, an NPE trained exclusively on samples from $p_s(\boldsymbol{x} \mid \boldsymbol{\theta})$ may provide accurate posterior estimates for simulated observations but become inaccurate or poorly calibrated when applied to real observations from $p_o(\boldsymbol{x} \mid \boldsymbol{\theta})$.

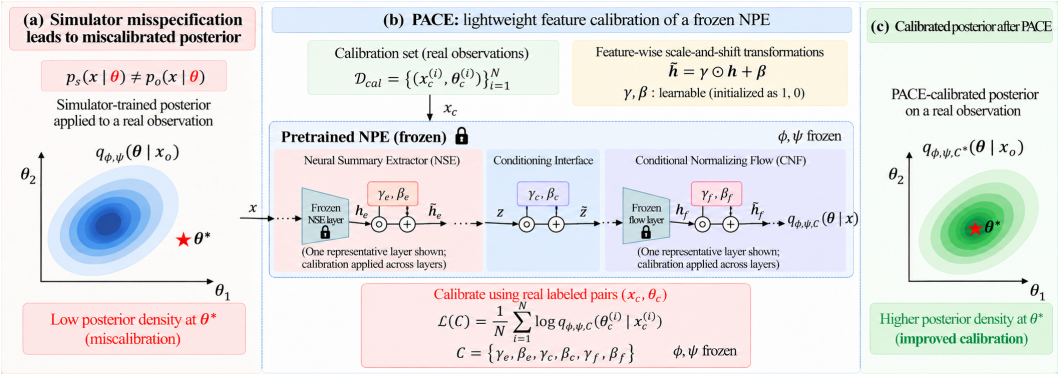


Figure 1: Simulator misscpecification can lead to miscalibrated posterior for real observations. PACE keeps the pretrained NPE frozen and learns a light weight scale and shift parameters which form calibration parameters using a small labeled calibration set. The Calibrated NPE model then performs amortised posterior inference on unseen real observations

We consider a setting in which simulated data are abundant, while only a limited number of labeled observations from the real system are available. Specifically, after pretraining the NPE on simulated parameter–observation pairs, we assume access to a **small calibration dataset** $\mathcal{D}_{\text{cal}} = \{(\theta_i, x_{c,i})\}_{i=1}^N$, where $x_{c,i}$ is a real observation associated with a known parameter value θ_i, and N is substantially smaller than the number of simulated samples used for pretraining. A straightforward strategy is to fine-tune the entire pretrained NPE using $\mathcal{D}_{\text{cal}}$. However, this requires adapting a potentially large number of model parameters using a comparatively small real calibration set. We therefore consider whether the simulator-pretrained estimator can instead be calibrated using a substantially smaller set of trainable parameters while retaining amortized posterior inference on previously unseen real observations. Thus, our setting assumes access to:

- a simulator $\mathcal{S}$ that generates simulated observations $x_s \sim p_s(\boldsymbol{x} \mid \boldsymbol{\theta})$ for parameters $\theta \sim p(\theta)$;
- a simulator-pretrained NPE $q_{\phi,\psi}(\boldsymbol{\theta} \mid \boldsymbol{x_s})$ learned from simulated parameter–observation pairs $(\boldsymbol{\theta}, \boldsymbol{x_s})$;
- a small labeled real calibration dataset $\mathcal{D}_{\text{cal}} = \{(\boldsymbol{\theta}_i, \boldsymbol{x}_{c,i})\}_{i=1}^N$; and
- previously unseen real observations $\boldsymbol{x}_o$ for which posterior inference over $\boldsymbol{\theta}$ is required.

3 How Little Adaptation Is Enough?

To investigate how little adaptation is sufficient to address simulator misspecification, we intentionally restrict the adaptation capacity of the pretrained NPE. The simulator-trained NSE parameters ψ and posterior estimator parameters ϕ remain completely frozen. Instead, we introduce only a small set of trainable parameters, which we refer to as *calibration parameters*. While different mechanisms can be used to adapt pretrained models, we deliberately consider lightweight feature-wise transformations to test whether a minimal recalibration of the representations learned from simulated data is sufficient to improve posterior inference on real observations.

3.1 Feature-wise Scale-and-Shift Calibration

We parameterize the calibration using feature-wise scale-and-shift transformations (Lian et al., 2022). Given a representation $h^{(\ell)} \in \mathbb{R}^{d_\ell}$ produced by a pretrained layer, we define

$$\widetilde{h}^{(\ell)} = \gamma^{(\ell)} \odot h^{(\ell)} + \beta^{(\ell)}, \tag{2}$$

where $\gamma^{(\ell)}, \beta^{(\ell)} \in \mathbb{R}^{d_\ell}$ are learnable feature-wise scale and shift parameters, respectively, and $\odot$ denotes element-wise multiplication. Thus, each feature can be independently rescaled and shifted while the pretrained transformation producing it remains fixed. For a d_ℓ-dimensional representation, Eq. equation 2 introduces only $2d_\ell$ trainable parameters.

As Illustrated in Figure 1, We introduce these transformations at three locations in the inference pathway: within the NSE, at the conditioning interface, and within the conditional normalizing flow. For the NSE, an intermediate representation is calibrated as

$$\widetilde{h}_e^{(\ell)} = \gamma_e^{(\ell)} \odot h_e^{(\ell)} + \beta_e^{(\ell)}, \qquad \ell = 1, \dots, L_e. \tag{3}$$

The final NSE representation $z = h_\psi(x)$ is further calibrated at the conditioning interface as

$$\widetilde{z} = \gamma_c \odot z + \beta_c, \tag{4}$$

before being provided as context to the posterior estimator. Similarly, intermediate representations within the conditional normalizing flow are calibrated as

$$\widetilde{h}_f^{(\ell)} = \gamma_f^{(\ell)} \odot h_f^{(\ell)} + \beta_f^{(\ell)}, \qquad \ell = 1, \dots, L_f. \tag{5}$$

Collectively, these parameters form the calibration parameter set

$$\mathcal{C} = \left\{ \{\gamma_e^{(\ell)}, \beta_e^{(\ell)}\}_{\ell=1}^{L_e}, \gamma_c, \beta_c, \{\gamma_f^{(\ell)}, \beta_f^{(\ell)}\}_{\ell=1}^{L_f} \right\}. \tag{6}$$

They are initialized with $\gamma = \mathbf{1}$ and $\beta = \mathbf{0}$, such that the calibrated model is identical to the simulator-trained NPE before calibration. Importantly, $\mathcal{C}$ constitutes the entire adaptation space; the pretrained parameters ψ and ϕ are never updated.

This construction is deliberately simple. By restricting adaptation to feature-wise calibration parameters, we test whether substantial modification of the simulator-trained NPE is necessary to address misspecification. If this restricted adaptation space is sufficient to improve posterior inference, it suggests that much of the inference structure learned from simulated data remains useful for real observations and can be corrected through lightweight calibration.

3.2 Calibration and Inference

Given the labeled real calibration set $\mathcal{D}_{\text{cal}} = \{(\boldsymbol{\theta_i}, \boldsymbol{x_{o,i}})\}_{i=1}^N$, we optimize only $\mathcal{C}$ using the same posterior objective used for neural posterior estimation:

$$\mathcal{C}^\star = \arg\min_{\mathcal{C}} -\frac{1}{N} \sum_{i=1}^{N} \log q_{\boldsymbol{\phi},\boldsymbol{\psi},\mathcal{C}}(\boldsymbol{\theta_i} \mid \boldsymbol{x_{c,i}}), \qquad \boldsymbol{\psi}, \boldsymbol{\phi} \text{ fixed}. \tag{7}$$

Thus, calibration directly optimizes posterior inference on the available real observations without introducing an additional alignment, discrepancy, or transport objective. Once $\mathcal{C}^\star$ has been learned, posterior inference for a previously unseen real observation $\boldsymbol{x_o}$ is performed as

$$q_{\boldsymbol{\phi},\boldsymbol{\psi},\mathcal{C}^\star}(\boldsymbol{\theta} \mid \boldsymbol{x_o}). \tag{8}$$

The learned calibration parameters are shared across observations and no optimization is performed at test time. PACE therefore retains the amortized inference property of the original NPE while adapting only a small calibration parameter space.

When is lightweight calibration sufficient? Consider the representation-calibration subfamily of PACE, where $z_s = h_\psi(x_s)$ and $z_o^C = T_C(h_\psi(x_o))$ with $T_C(z) = \gamma \odot z + \beta$. Let $p_s^z(\cdot \mid \theta)$ and $p_o^{z,C}(\cdot \mid \theta)$ denote the conditional distributions over these representations induced by $x_s \sim p_s(x \mid \theta)$ and $x_o \sim p_o(x \mid \theta)$, respectively. Suppose there exists $C^\star$ such that $z_o^{C^\star}$ is sufficient for θ under the target process and $p_o^{z,C^\star}(\cdot \mid \theta) = p_s^z(\cdot \mid \theta)$ for all θ. If the simulator-pretrained posterior estimator is exact in representation space, $q_\phi(\theta \mid z) = p_s(\theta \mid z)$, and the simulator and target domains share the same parameter prior, then

$$q_\phi(\theta \mid T_{C^\star}(h_\psi(x_o))) = p_o(\theta \mid x_o).$$

Thus, if misspecification can be corrected by an affine transformation of a sufficient simulator-trained representation, a PACE subfamily already contains the exact target posterior. This is a sufficient-condition/realizability result, not an assumption that arbitrary misspecification is affine; full details are given in Appendix D.

Interpretation. The proposition formalizes the regime targeted by PACE: lightweight adaptation can suffice when simulator pretraining preserves parameter-relevant information and the remaining simulator-to-target discrepancy is low-complexity.

4 RELATED WORK

Addressing model misspecification has recently become an active area of research in simulation based inference (SBI). Several approaches within neural SBI have been proposed to mitigate simulator misspecification. RNPE (Ward et al., 2022) introduces an explicit probabilistic model between real and simulated summary statistics. Huang et al. (Huang et al., 2023) instead learn summary statistics that are less sensitive to misspecification by regularising the mismatch between simulated and observed data. While Huang et al. (Huang et al., 2023) propose training time correction, recently, Khoo et al. (Khoo et al., 2026) propose a test-time correction of summary statistics using maximum mean discrepancy (MMD). A complimentary line of work uses a real calibration data to learn the discrepancy among simulated and real samples. ROPE Wehenkel et al. (2025) learns representation of simulator and real observations and uses optimal transport to model the discrepancy, which is subsequently used to correct the posterior at the infernce time. However, this correction is transductive and requires access to test observations during inference. FRISBI (Senouf et al., 2025) addresses this transductive limitation by learning an inductive correction through joint optimal-transport alignment and supervised calibration, and subsequently amortizes the OT-induced posterior using a conditional normalizing flow. FMCPE (Ruhlmann et al., 2025) first learns a posterior estimator from simulated data and subsequently learns a flow-matching transport from real calibration data to correct its posterior predictions. These approaches raise a natural question: **does correcting simulator misspecification necessarily require a sophisticated correction mechanism?**. This has been the motivation our work which investigates whether a substantially smaller calibration space can be sufficient for adapting simulator-trained posterior inference.

Parameter-efficient adaptation arises as a natural way to accomplish this. Instead of updating all the parameters of a pretrained model, these methods introduce or modify only a small number of trainable parameters while keeping the pretrained model largely fixed. Popular adaptation methods include adapters (Houlsby et al., 2019), low-rank adaptation (Hu et al., 2022), and feature-wise transformations (Perez et al., 2018; Lian et al., 2022). Among these, feature-wise scaling and shifting provides an extremely lightweight and flexible mechanism for modifying learned representations with no need to update the underlying pretrained transformations (Lian et al., 2022). Motivated by the success of these methods in settings such as transfer and continual learning, PACE explores whether posterior inference under simulator misspecification can similarly be calibrated within a compact parameter space, rather than relying on explicit discrepancy modeling, distribution matching alignment, or optimal transport. The advantage of PACE lies in leveraging the representations and posterior structure learned from abundant simulated data while calibrating the estimator to real observations in an parameter-efficient manner. These parameters are learned from limited labeled real data and optimized directly through the posterior objective, while the simulator-pretrained model remains frozen. Once calibrated, the same estimator can be applied directly to unseen real observations, preserving amortized posterior inference.

5 EXPERIMENTS: IS LIGHTWEIGHT CALIBRATION SUFFICIENT?

5.1 BENCHMARK DATASETS

We evaluate PACE under several forms of simulator misspecification, ranging from controlled discrepancies in synthetic systems to gaps between analytic simulators and physical measurements. Following RoPE Wehenkel et al. (2025) and FRISBI Senouf et al. (2025), we consider three synthetic tasks: **CS**, **SIR**, and **Pendulum**, and two real-world tasks, **Light Tunnel** and **Wind Tunnel**, from Causal Chambers (Gamella et al., 2025). **CS.** The Cancer and Stromal Cell Simulator models cancer and stromal cell growth in 2D, controlled by $(\lambda_c, \lambda_p, \lambda_d)$. It tracks the total number of cells and the distance of stromal cells from their nearest cancer cell. Misspecification is introduced by deleting newly generated cancer cells that are too close to their parent cells. **SIR.** The SIR task models epidemics using infection (β) and recovery (γ) rates. Its observations capture the mean, median, and peak infections, days of peak and half-total infections, and lag-1 autocorrelation. Misspecification delays 5% of weekend cases to Monday. **Pendulum.** The base simulator models the horizontal position of a frictionless pendulum using its fundamental frequency (ω_0) and amplitude (A), with randomness from a phase shift and measurement noise; real observations are generated from a damped pendulum to introduce friction misspecification. **Light Tunnel.** This real-world task uses a camera to capture a controllable light source viewed through two rotating polarizers. The objective

is to infer the RGB settings and dimming effect (α) from the captured images. The simulator uses $\theta := [R, G, B, \alpha]$ to generate synthetic images rendered with adjusted color $[\alpha R, \alpha G, \alpha B]$. **Wind Tunnel.** This task uses a physical chamber with controllable fans, barometers, and a hatch regulating an external opening. Internal air-pressure curves are generated by applying a brief impulse to the intake fan, and the objective is to infer the hatch position, $\theta := H \in [0, 45]$, from a pressure curve.

5.2 Baselines

We compare PACE against several baselines for adapting posterior inference under simulator misspecification. For a fair comparison, all methods use the same task-specific NPE architecture and calibration split. **Full fine-tuning (FT)** updates all parameters of the pretrained NPE. **J-NPE** trains from scratch using minibatches containing equal numbers of simulated and target pairs. **RoPE** fine-tunes the encoder using paired simulations and then applies semi-balanced entropic optimal transport over a fixed simulation bank to construct a transductive posterior mixture. **FRISBI** combines supervised and transport-based encoder alignment and distills the resulting mixture into a new conditional flow; we use its full two-stage variant. All checkpoints are selected using only the calibration validation data.

We additionally report three fixed references. The **Prior** ignores the observation. The **Base NPE** is trained only on simulations and applied directly to real observations without adaptation, thereby measuring the effect of simulator misspecification. **Oracle SBI** uses the same task-specific NPE architecture but is trained on labeled real data. It serves as an upper reference and uses substantially more real data than the calibration baselines.

We use labeled target-domain calibration data to train the calibration parameters of PACE and the corresponding adaptation components of each baseline. We consider calibration-set sizes $n \in \{50, 100, 200, 500, 1000\}$, with sets nested within each seed. For each calibration size, the available data are split into training and validation sets, with the validation split used for model selection and early stopping. All methods are evaluated on the same disjoint target-domain test set. We report the mean and sample standard deviation over three calibration seeds.

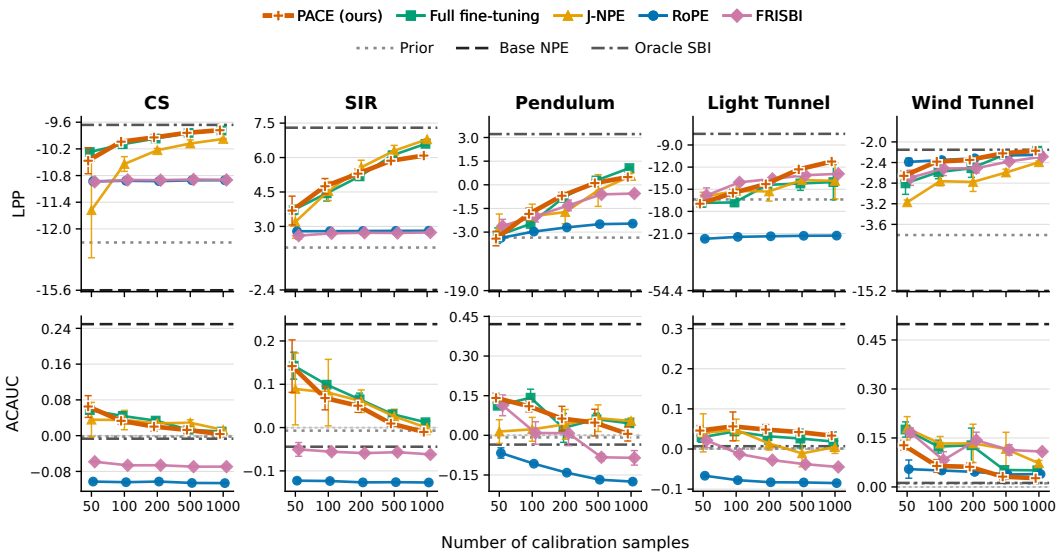


Figure 2: Results on Synthetic and Physical tasks.

5.3 Evaluation Metrics

We report **log posterior probability (LPP)**, posterior-mean normalized root mean squared error **(NRMSE)**, **average coverage error**, area calibration **ACAUC**, **energy score**, and joint classifier two-sample test accuracy (**jC2ST**). LPP assesses the density assigned to the true parameter, NRMSE measures point-estimation accuracy, and the remaining quantities assess calibration and sample quality. Appendix A defines every metric in detail.

6 RESULTS

Figures 2 shows the results on synthetic and physical tasks. Across all five tasks, PACE consistently repairs the large loss caused by applying the simulation-trained NPE directly to real observations. More details about implementation is given in the appendix C.

Synthetic tasks. On CS, PACE attains competitive LPP from $n = 100$ onward and its ACAUC steadily approaches zero as the calibration set grows. On SIR, J-NPE obtains the largest LPP at the highest budgets, whereas PACE remains competitive and provides ACAUC close to zero. Pendulum presents the clearest data-dependent ranking. PACE performs particularly well at intermediate budgets, while full fine-tuning becomes stronger with $n = 500$ to 1000. RoPE and FRISBI can be well calibrated on individual tasks, but their lower LPP illustrates that a favorable ACAUC need not imply an informative posterior. PACE instead offers a favorable compromise between informativeness and calibration.

Physical tasks. The same pattern transfers to real world physical task. On Light Tunnel, PACE improves markedly with additional calibration data and achieves the strongest LPP among the adaptation methods at the two largest budgets, while retaining modest ACAUC. On Wind Tunnel, RoPE is strongest at the smallest budgets, but PACE catches up in LPP and reaches the leading group by $n = 500$, with ACAUC decreasing toward zero. Thus, PACE remains effective for both high-dimensional images and pressure time series without introducing a task-specific adaptation mechanism.

These results are notable because PACE does not train a new posterior model or modify all pretrained weights. It freezes the NPE and learns only identity-initialized feature-wise scales and shifts with a context adapter. In contrast, the baselines require unrestricted fine-tuning, joint simulation–real training, a transductive optimal-transport solve, or a two-stage alignment and distillation procedure. The experiments therefore support our central claim that small affine corrections to a pretrained posterior estimator can recover much of the performance of substantially more involved adaptation pipelines. Complete NRMSE, coverage, energy-score, and jC2ST results are reported in the appendix tables.

Computational cost. PACE requires only one adaptation stage and uses at most 3.19% of the base NPE parameters. Full fine tuning and J-NPE optimize a complete NPE, while FRISBI adapts an encoder and then trains a conditional flow in a second stage. RoPE only optimizes the encoder, but its cost depends strongly on the encoder size and posterior evaluation requires a transport weighted mixture over the cached simulation bank. PACE retains ordinary amortized NPE inference and stores only the learned scales and shifts for each target domain.

6.1 ABLATIONS: HOW MUCH CALIBRATION IS ACTUALLY NECESSARY?

Variant	LPP ↑	NRMSE ↓	Coverage MAE ↓	ACAUC (0)	Energy ↓	jC2ST (0.5)
Full PACE	$\mathbf{-0.695 \pm 0.228}$	$\mathbf{0.318 \pm 0.020}$	0.078 ± 0.051	0.063 ± 0.046	$\mathbf{0.230 \pm 0.020}$	$\mathbf{0.553 \pm 0.025}$
Encoder + context (γ, β)	-1.257 ± 0.129	0.363 ± 0.014	0.051 ± 0.017	-0.063 ± 0.028	0.282 ± 0.009	0.586 ± 0.029
Flow + context (γ, β)	-1.782 ± 0.057	0.489 ± 0.019	$\mathbf{0.014 \pm 0.005}$	-0.004 ± 0.009	0.407 ± 0.021	0.612 ± 0.005
All layers, scale only (γ)	$\underline{-0.923 \pm 0.179}$	$\underline{0.348 \pm 0.016}$	0.065 ± 0.032	0.045 ± 0.037	$\underline{0.262 \pm 0.023}$	$\underline{0.564 \pm 0.016}$
All layers, shift only (β)	-1.104 ± 0.060	0.392 ± 0.016	$\underline{0.035 \pm 0.017}$	-0.001 ± 0.030	0.307 ± 0.016	0.582 ± 0.029

Table 1: PACE ablations on Pendulum with $n = 200$; γ and β denote scale and shift, respectively.

The Pendulum ablations at $n = 200$ (Table 1) show that adapting the encoder is more effective than adapting the flow alone, indicating that correcting the learned observation representation is particularly important. Scale-only adaptation retains most of the LPP gain, but the full scale-and-shift model achieves the best results across all the metrics. These results support distributing small affine corrections across the pretrained NPE instead of relying on a single adapter location or transformation.

7 ROBUSTNESS TO PRIOR MISSPECIFICATION

We study prior misspecification on Pendulum while keeping the pretrained NPE fixed. The NPE is trained under the broad assumed prior $p_A(\boldsymbol{\theta}) = \mathcal{U}([0, 3] \times [0.5, 10])$. Calibration and test parameters

are instead sampled from $p_\lambda(\boldsymbol{\theta}) = (1-\lambda)p_A(\boldsymbol{\theta}) + \lambda p_B(\boldsymbol{\theta})$ where $p_B(\theta) = \mathcal{U}([0,1] \times [0.5,3])$, $\lambda \in \{0, 0.1, 0.2, 0.3, 0.4\}$. Thus $\lambda = 0$ recovers the assumed simulation prior, while larger values place an increasing fraction of the target probability mass on the narrower prior p_B. For every value of λ, PACE adapts the same pretrained NPE using $n = 500$ labeled target pairs. We report the mean and sample standard deviation over three seeds.

Figure 3 shows that PACE remains stable as the prior discrepancy increases. Its mean LPP varies only from 0.127 at $\lambda = 0$ to -0.113 at $\lambda = 0.4$, and its mean ACAUC remains between -0.004 and 0.064 throughout the sweep. PACE also maintains a large LPP improvement over the frozen NPE and both prior references for every tested value of λ. These results indicate that the learned affine corrections are robust to the tested changes in prior mass and do not rely on exact agreement between the simulation and target priors.

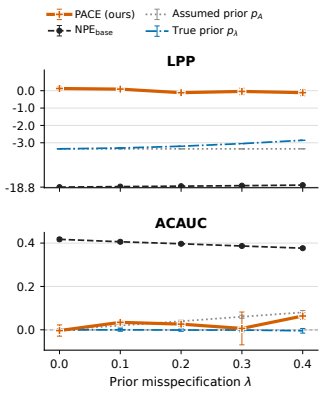


Figure 3: Pendulum prior misspecification results.

8 Degree of PACE calibration updates versus scale of misspecification

PACE uses identity and zero initialization for scale and shift parameters, we expect the update of calibration parameters according to scale of misspecification. If degree of misspsecification is small the calibration parameters should not deviate much from identity and zero. On the other hand scale and shift parameter shows larger updates with higer degree of misspecification In Figure 4 we show this on Pendulum task by varying the the value of damping (α) and measuring the change in scale and sift parameters and LPP.

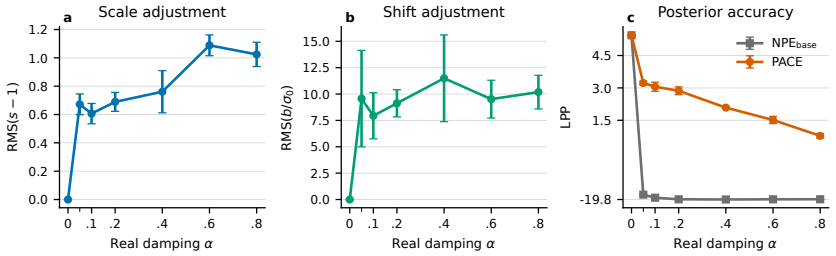


Figure 4: Variation in calibrated scale and shift parameters across various misspecification settings.

9 Conclusion

We introduce PACE, a parameter-efficient calibration framework for posterior inference under simulator misspecification. Rather than introducing an explicit discrepancy model or distribution alignment objective or multistage procedure, PACE explores how much of simulator trained inference need to change. It adapts only a small set of feature-wise scale and shift parameters from limited labeled real data,directly through the posterior objective, while the simulator trained NPE is kept frozen.

Across various synthetic and real benchmarks PACE achieves performance better than or comparable to SOTA approaches, while introducing only 0.5–4% additional parameters. Our results across different calibration-set sizes and calibration parameter spaces show that substantial improvements in posterior inference can be achieved while preserving most of the simulator-trained model. Importantly, PACE retains fast amortized inference on unseen real observations without test-time optimization or additional simulator evaluations. Our results suggest that when simulator-trained representations retain the information needed for posterior inference, correcting simulator misspecification may not require re-learning the inference model or explicitly aligning simulated and real distributions; a lightweight calibration of the existing inference pathway can be sufficient.

10 Impact Statement

AI use statement

In this work, we used generative AI tools to aid or polish writing. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

This work is methodological and does not involve human subjects or sensitive personal data. Our experiments are based on synthetic benchmarks and previously released real-world datasets. We do not identify any direct ethical concerns related to the proposed methodology. As with other simulation-based inference methods, its utility in downstream applications should consider the suitability of the simulator, data, and modeling assumptions.

Reproducibility statement

We provide detailed descriptions of the experimental setup, model architectures, training and calibration procedures, hyperparameters, evaluation metrics, and implementation details in the Appendix for reproducibilty. We will share code at a later stage.

References

Patrick Cannon, Daniel Ward, and Sebastian M. Schmon. Investigating the impact of model misspecification in neural simulation-based inference. *arXiv preprint arXiv:2209.01845*, 2022.

Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020. doi: 10.1073/pnas.1912789117.

Michael Deistler, Jan Boelts, Peter Steinbach, Guy Moss, Thomas Moreau, Manuel Gloeckler, Pedro L. C. Rodrigues, Julia Linhart, Janne K. Lappalainen, Benjamin Kurt Miller, Pedro J. Gonçalves, Jan-Matthis Lueckmann, Cornelius Schröder, and Jakob H. Macke. Simulation-based inference: A practical guide, 2025.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Juan L. Gamella, Jonas Peters, and Peter Bühlmann. Causal chambers as a real-world physical testbed for ai methodology. *Nature Machine Intelligence*, 7:107–118, 2025. doi: 10.1038/s42256-024-00964-x.

Joeri Hermans, Volodimir Begy, and Gilles Louppe. Likelihood-free mcmc with amortized approximate ratio estimators. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4239–4248. PMLR, 2020.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Daolang Huang, Ayush Bharti, Amauri Souza, Luigi Acerbi, and Samuel Kaski. Learning robust statistics for simulation-based inference under model misspecification. In *Advances in Neural Information Processing Systems*, volume 36, pp. 7289–7310, 2023.

Sherman Khoo, Dennis Prangle, Song Liu, and Mark A. Beaumont. Minimum distance summaries for robust neural posterior estimation. In *Proceedings of the 43rd International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2026.

Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

George Papamakarios and Iain Murray. Fast ϵ-free inference of simulation models with bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

George Papamakarios, David Sterratt, and Iain Murray. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 837–848. PMLR, 2019.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Pierre-Louis Ruhlmann, Pedro L. C. Rodrigues, Michael Arbel, and Florence Forbes. Flow matching for robust simulation-based inference under model misspecification. *arXiv preprint arXiv:2509.23385*, 2025.

Marvin Schmitt, Paul-Christian Bürkner, Ullrich Köthe, and Stefan T. Radev. Detecting model misspecification in amortized bayesian inference with neural networks. In *DAGM German Conference on Pattern Recognition*, pp. 541–557. Springer, 2023.

Ortal Senouf, Antoine Wehenkel, Cédric Vincent-Cuaz, Emmanuel Abbe, and Pascal Frossard. Inductive domain transfer in misspecified simulation-based inference. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=PhnquAa8eV`.

Daniel Ward, Patrick Cannon, Mark Beaumont, Matteo Fasiolo, and Sebastian M. Schmon. Robust neural posterior estimation and statistical model criticism. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Antoine Wehenkel, Juan L. Gamella, Ozan Sener, Jens Behrmann, Guillermo Sapiro, Marco Cuturi, and Jörn-Henrik Jacobsen. Addressing misspecification in simulation-based inference through data-driven calibration. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 65949–65980. PMLR, 2025.

A Evaluation Metrics

Let $\{(\boldsymbol{\theta}_i, \mathbf{x}_i)\}_{i=1}^N$ be the held-out target-domain test set, let $q(\boldsymbol{\theta} \mid \mathbf{x})$ denote an estimated posterior, and let $\boldsymbol{\theta}_{ij} \sim q(\cdot \mid \mathbf{x}_i)$ be its jth posterior draw. We use M posterior draws per test observation and K parameters.

Log posterior probability (LPP). We measure the average log density assigned to the ground-truth parameter,

$$\mathrm{LPP} = \frac{1}{N} \sum_{i=1}^{N} \log q(\boldsymbol{\theta}_i \mid \mathbf{x}_i). \tag{9}$$

LPP is computed in the physical parameter coordinates. It is therefore comparable across methods within a task, but not across tasks with different parameterizations. Higher values are better.

Posterior-mean error. For parameter k, physical-unit RMSE is

$$\mathrm{RMSE}_k = \left[\frac{1}{N} \sum_{i=1}^{N} (\widehat{\mu}_{ik} - \theta_{ik})^2 \right]^{1/2}, \qquad \widehat{\boldsymbol{\mu}}_i = \frac{1}{M} \sum_{j=1}^{M} \boldsymbol{\theta}_{ij}. \tag{10}$$

To aggregate parameters with different units, we divide each error by the prior standard deviation s_k and report

$$\mathrm{NRMSE} = \left[\frac{1}{NK} \sum_{i=1}^{N} \sum_{k=1}^{K} \left(\frac{\widehat{\mu}_{ik} - \theta_{ik}}{s_k} \right)^2 \right]^{1/2}. \tag{11}$$

Lower RMSE and NRMSE are better.

Empirical coverage. For nominal level a, let $[L_{ik}^{(a)}, U_{ik}^{(a)}]$ be the central posterior credible interval for parameter k. Its empirical coverage is

$$C_k(a) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left\{ L_{ik}^{(a)} \le \theta_{ik} \le U_{ik}^{(a)} \right\}. \tag{12}$$

We evaluate $a \in \{0.50, 0.80, 0.90, 0.95\}$. The reported coverage gap is $a - C_k(a)$, so positive values indicate undercoverage. Coverage MAE averages $|a - C_k(a)|$ over levels and parameters, while coverage bias averages the signed gap. Values closer to zero are better.

ACAUC. For every test pair and parameter, define the posterior probability integral transform

$$u_{ik} = \frac{1}{M} \sum_{j=1}^{M} \mathbb{1}\{\theta_{ijk} \le \theta_{ik}\}. \tag{13}$$

Our signed area calibration statistic is

$$\mathrm{ACAUC}_k = \frac{1}{N} \sum_{i=1}^{N} |2u_{ik} - 1| - \frac{1}{2}, \qquad \mathrm{ACAUC} = \frac{1}{K} \sum_{k=1}^{K} \mathrm{ACAUC}_k. \tag{14}$$

A calibrated posterior has uniformly distributed PIT values and hence ACAUC near zero. Positive and negative values distinguish different deviations from uniformity; performance is judged by distance to zero.

Energy score. We standardize every parameter by its prior standard deviation and write $\widetilde{\boldsymbol{\theta}}_{ij} = \boldsymbol{\theta}_{ij} \oslash \mathbf{s}$. With two independent posterior draws $\widetilde{\boldsymbol{\theta}}_i$ and $\widetilde{\boldsymbol{\theta}}'_i$, the energy score is estimated as

$$\mathrm{ES} = \frac{1}{N} \sum_{i=1}^{N} \left[\mathbb{E} \left\| \widetilde{\boldsymbol{\theta}}_i - \widetilde{\boldsymbol{\theta}}_i^{\star} \right\|_2 - \frac{1}{2} \mathbb{E} \left\| \widetilde{\boldsymbol{\theta}}_i - \widetilde{\boldsymbol{\theta}}'_i \right\|_2 \right], \tag{15}$$

where $\widetilde{\boldsymbol{\theta}}_i^{\star}$ is the standardized true parameter. The expectations are Monte Carlo averages over posterior draws. Lower values are better.

Joint classifier two-sample test (jC2ST). We compare samples from the held-out target joint distribution $p(\boldsymbol{\theta}, \mathbf{x})$ with samples from $q(\boldsymbol{\theta} \mid \mathbf{x})p(\mathbf{x})$. For each test observation, one true parameter and one posterior draw are paired with the same observation. A binary classifier is trained to distinguish the two joint samples using three-fold cross-validation over observation indices. We report mean held-out classification accuracy. An accuracy of 0.5 indicates that the two joint distributions are indistinguishable; larger departures from 0.5 indicate posterior error.

B FULL RESULTS

In this section, we present the complete results for all tasks. Table 2, 3 and 4 sshow the results for the synthetic tasks **CS**, **SIR**, and **Pendulum**, respectively. Table 5 and 6 show the full results for the real-world physical tasks **Light Tunnel** and **Wind Tunnel**, respectively.

Method	n	LPP ↑	NRMSE ↓	Coverage MAE ↓	ACAUC (0 ideal)	Energy ↓	JC2ST (0.5 ideal)
Prior	–	-12.306	0.996	0.007	-0.002	1.140	0.854
NPE_{base}	–	-15.611	0.981	0.344	0.249	1.190	0.838
SBI (oracle)	–	-9.660	0.653	0.010	-0.007	0.707	0.497
PACE	50	-10.467 ± 0.292	0.705 ± 0.020	0.083 ± 0.034	0.065 ± 0.024	0.774 ± 0.025	0.606 ± 0.009
	100	-10.038 ± 0.034	0.676 ± 0.001	0.040 ± 0.008	0.033 ± 0.008	0.737 ± 0.001	0.566 ± 0.011
	200	-9.942 ± 0.057	0.671 ± 0.007	0.028 ± 0.013	0.020 ± 0.011	0.730 ± 0.008	0.554 ± 0.016
	500	-9.838 ± 0.041	0.662 ± 0.004	0.016 ± 0.009	0.013 ± 0.008	0.718 ± 0.005	0.532 ± 0.015
	1000	-9.780 ± 0.015	0.657 ± 0.002	0.008 ± 0.004	0.004 ± 0.006	0.712 ± 0.002	0.515 ± 0.011
RoPE	50	-10.929 ± 0.029	0.753 ± 0.008	0.071 ± 0.002	-0.103 ± 0.003	0.832 ± 0.010	0.677 ± 0.008
	100	-10.917 ± 0.012	0.750 ± 0.004	0.071 ± 0.001	-0.104 ± 0.002	0.829 ± 0.004	0.682 ± 0.012
	200	-10.926 ± 0.011	0.751 ± 0.002	0.071 ± 0.000	-0.103 ± 0.001	0.831 ± 0.002	0.688 ± 0.005
	500	-10.906 ± 0.007	0.749 ± 0.003	0.071 ± 0.001	-0.106 ± 0.001	0.828 ± 0.004	0.682 ± 0.006
	1000	-10.905 ± 0.001	0.749 ± 0.001	0.071 ± 0.001	-0.106 ± 0.001	0.828 ± 0.001	0.680 ± 0.009
FRISBI	50	-10.941 ± 0.028	0.770 ± 0.009	0.053 ± 0.002	-0.058 ± 0.001	0.852 ± 0.011	0.668 ± 0.004
	100	-10.899 ± 0.023	0.764 ± 0.007	0.058 ± 0.003	-0.066 ± 0.002	0.844 ± 0.009	0.670 ± 0.005
	200	-10.904 ± 0.014	0.767 ± 0.004	0.056 ± 0.001	-0.066 ± 0.003	0.848 ± 0.006	0.671 ± 0.003
	500	-10.889 ± 0.008	0.765 ± 0.001	0.059 ± 0.004	-0.069 ± 0.000	0.846 ± 0.002	0.667 ± 0.006
	1000	-10.898 ± 0.003	0.765 ± 0.002	0.058 ± 0.003	-0.069 ± 0.004	0.846 ± 0.002	0.671 ± 0.012
J-NPE	50	-11.578 ± 1.097	0.833 ± 0.144	0.059 ± 0.028	0.036 ± 0.039	0.938 ± 0.182	0.715 ± 0.121
	100	-10.540 ± 0.166	0.704 ± 0.012	0.050 ± 0.020	0.035 ± 0.022	0.775 ± 0.015	0.616 ± 0.018
	200	-10.230 ± 0.041	0.703 ± 0.004	0.036 ± 0.006	0.028 ± 0.004	0.770 ± 0.005	0.610 ± 0.007
	500	-10.078 ± 0.080	0.699 ± 0.005	0.035 ± 0.006	0.029 ± 0.007	0.765 ± 0.007	0.571 ± 0.010
	1000	-9.977 ± 0.012	0.694 ± 0.003	0.017 ± 0.003	0.013 ± 0.003	0.758 ± 0.003	0.584 ± 0.008
FT	50	-10.272 ± 0.089	0.699 ± 0.021	0.073 ± 0.009	0.057 ± 0.006	0.766 ± 0.024	0.596 ± 0.016
	100	-10.094 ± 0.078	0.676 ± 0.008	0.055 ± 0.009	0.044 ± 0.009	0.738 ± 0.012	0.568 ± 0.015
	200	-9.968 ± 0.043	0.673 ± 0.009	0.042 ± 0.010	0.034 ± 0.006	0.733 ± 0.011	0.561 ± 0.024
	500	-9.841 ± 0.027	0.659 ± 0.002	0.017 ± 0.005	0.013 ± 0.005	0.715 ± 0.002	0.531 ± 0.007
	1000	-9.787 ± 0.013	0.657 ± 0.001	0.013 ± 0.003	0.009 ± 0.004	0.712 ± 0.001	0.525 ± 0.009

Table 2: CS results. Mean ± sample standard deviation over three calibration seeds.

C IMPLEMENTATION DETAILS

Shared training procedure Every NPE consists of an observation encoder followed by a conditional neural spline flow. We use eight spline bins and ReLU conditioner networks. The NPE is trained with Adam by minimizing the negative log posterior density on simulated parameter and observation pairs. We retain the checkpoint with the lowest validation negative log likelihood. PACE freezes every NPE parameter and places a feature wise affine map $h \mapsto s \odot h + b$ after each eligible linear or convolutional layer in the encoder and flow conditioner. The encoder output that conditions the flow is adapted as the context representation. All scales are initialized to one and all shifts to zero, so adaptation starts from exactly the pretrained NPE. Only the scale and shift parameters are optimized with Adam using the same posterior negative log likelihood on labeled target pairs. For every target data size, 80 percent of the calibration pairs are used for optimization and 20 percent are used for validation. The reported calibration sizes are 50, 100, 200, 500, and 1000. Early stopping and checkpoint selection use only the validation negative log likelihood.

Method	n	LPP ↑	NRMSE ↓	Coverage MAE ↓	ACAUC (0 ideal)	Energy ↓	JC2ST (0.5 ideal)
Prior	–	2.079	0.989	0.008	-0.007	0.869	0.885
NPE_{base}	–	-2.411	0.614	0.356	0.239	0.455	0.766
SBI (oracle)	–	7.300	0.263	0.044	-0.044	0.165	0.513
PACE	50	3.691 ± 0.628	0.523 ± 0.016	0.205 ± 0.075	0.142 ± 0.061	0.316 ± 0.016	0.573 ± 0.016
	100	4.750 ± 0.336	0.491 ± 0.013	0.109 ± 0.033	0.068 ± 0.027	0.287 ± 0.010	0.563 ± 0.007
	200	5.300 ± 0.208	0.490 ± 0.013	0.077 ± 0.020	0.050 ± 0.015	0.280 ± 0.009	0.556 ± 0.004
	500	5.862 ± 0.054	0.461 ± 0.017	0.029 ± 0.004	0.009 ± 0.006	0.260 ± 0.009	0.540 ± 0.013
	1000	6.086 ± 0.090	0.444 ± 0.011	0.013 ± 0.002	-0.010 ± 0.005	0.251 ± 0.008	0.543 ± 0.015
RoPE	50	2.789 ± 0.010	0.714 ± 0.007	0.092 ± 0.003	-0.123 ± 0.005	0.590 ± 0.005	0.808 ± 0.005
	100	2.793 ± 0.009	0.705 ± 0.005	0.094 ± 0.001	-0.123 ± 0.002	0.585 ± 0.004	0.806 ± 0.001
	200	2.800 ± 0.007	0.703 ± 0.006	0.095 ± 0.001	-0.127 ± 0.002	0.582 ± 0.004	0.807 ± 0.003
	500	2.808 ± 0.009	0.701 ± 0.002	0.096 ± 0.001	-0.126 ± 0.001	0.582 ± 0.001	0.815 ± 0.002
	1000	2.811 ± 0.003	0.699 ± 0.003	0.096 ± 0.002	-0.127 ± 0.002	0.579 ± 0.002	0.811 ± 0.002
FRISBI	50	2.600 ± 0.151	0.774 ± 0.020	0.041 ± 0.009	-0.050 ± 0.016	0.654 ± 0.015	0.837 ± 0.002
	100	2.700 ± 0.022	0.768 ± 0.008	0.041 ± 0.009	-0.056 ± 0.006	0.648 ± 0.006	0.838 ± 0.002
	200	2.724 ± 0.020	0.756 ± 0.006	0.045 ± 0.004	-0.059 ± 0.003	0.639 ± 0.006	0.838 ± 0.005
	500	2.714 ± 0.009	0.761 ± 0.005	0.043 ± 0.004	-0.057 ± 0.005	0.643 ± 0.005	0.842 ± 0.007
	1000	2.738 ± 0.008	0.760 ± 0.003	0.047 ± 0.002	-0.062 ± 0.002	0.641 ± 0.003	0.832 ± 0.005
J-NPE	50	3.196 ± 0.709	0.615 ± 0.089	0.121 ± 0.125	0.089 ± 0.083	0.465 ± 0.091	0.680 ± 0.071
	100	4.542 ± 0.412	0.484 ± 0.199	0.113 ± 0.097	0.081 ± 0.077	0.356 ± 0.183	0.626 ± 0.099
	200	5.573 ± 0.310	0.423 ± 0.080	0.073 ± 0.034	0.061 ± 0.026	0.267 ± 0.040	0.572 ± 0.005
	500	6.297 ± 0.233	0.332 ± 0.012	0.035 ± 0.011	0.026 ± 0.017	0.211 ± 0.014	0.539 ± 0.014
	1000	6.793 ± 0.019	0.300 ± 0.011	0.028 ± 0.008	-0.001 ± 0.015	0.189 ± 0.006	0.538 ± 0.004
FT	50	3.722 ± 0.204	0.524 ± 0.014	0.204 ± 0.044	0.143 ± 0.031	0.322 ± 0.009	0.576 ± 0.010
	100	4.443 ± 0.338	0.493 ± 0.011	0.142 ± 0.008	0.099 ± 0.001	0.292 ± 0.010	0.567 ± 0.014
	200	5.180 ± 0.142	0.507 ± 0.024	0.098 ± 0.020	0.065 ± 0.014	0.289 ± 0.012	0.563 ± 0.005
	500	6.099 ± 0.167	0.385 ± 0.018	0.049 ± 0.010	0.031 ± 0.008	0.224 ± 0.008	0.527 ± 0.010
	1000	6.597 ± 0.054	0.332 ± 0.059	0.024 ± 0.008	0.012 ± 0.005	0.199 ± 0.016	0.529 ± 0.009

Table 3: SIR results. Mean ± sample standard deviation over three calibration seeds.

Method	n	LPP ↑	NRMSE ↓	Coverage MAE ↓	ACAUC (0 ideal)	Energy ↓	JC2ST (0.5 ideal)
Prior	–	-3.350	0.989	0.010	-0.008	0.893	0.810
NPE_{base}	–	-19.013	1.284	0.645	0.421	1.494	0.857
SBI (oracle)	–	3.217	0.148	0.036	-0.035	0.064	0.507
PACE	50	-2.992 ± 0.266	0.467 ± 0.037	0.159 ± 0.052	0.118 ± 0.048	0.405 ± 0.026	0.620 ± 0.031
	100	-1.846 ± 0.376	0.370 ± 0.014	0.141 ± 0.031	0.110 ± 0.022	0.289 ± 0.018	0.584 ± 0.027
	200	-0.695 ± 0.228	0.318 ± 0.020	0.078 ± 0.051	0.063 ± 0.046	0.230 ± 0.020	0.553 ± 0.025
	500	0.109 ± 0.188	0.272 ± 0.007	0.076 ± 0.034	0.048 ± 0.051	0.183 ± 0.009	0.535 ± 0.014
	1000	0.491 ± 0.134	0.263 ± 0.006	0.042 ± 0.006	0.006 ± 0.027	0.167 ± 0.008	0.528 ± 0.009
RoPE	50	-3.365 ± 0.111	0.410 ± 0.013	0.087 ± 0.018	-0.067 ± 0.020	0.323 ± 0.009	0.572 ± 0.008
	100	-2.966 ± 0.065	0.345 ± 0.012	0.086 ± 0.007	-0.107 ± 0.004	0.269 ± 0.009	0.564 ± 0.012
	200	-2.701 ± 0.040	0.322 ± 0.017	0.092 ± 0.004	-0.141 ± 0.003	0.244 ± 0.008	0.552 ± 0.010
	500	-2.495 ± 0.029	0.297 ± 0.004	0.107 ± 0.003	-0.168 ± 0.002	0.224 ± 0.003	0.552 ± 0.001
	1000	-2.460 ± 0.023	0.290 ± 0.002	0.110 ± 0.003	-0.175 ± 0.006	0.220 ± 0.002	0.547 ± 0.011
FRISBI	50	-2.624 ± 0.432	0.523 ± 0.024	0.156 ± 0.036	0.114 ± 0.039	0.437 ± 0.048	0.620 ± 0.052
	100	-1.993 ± 0.363	0.473 ± 0.069	0.100 ± 0.048	0.009 ± 0.043	0.373 ± 0.056	0.582 ± 0.024
	200	-1.307 ± 0.340	0.351 ± 0.016	0.111 ± 0.019	0.008 ± 0.047	0.270 ± 0.019	0.560 ± 0.024
	500	-0.607 ± 0.103	0.287 ± 0.014	0.093 ± 0.003	-0.083 ± 0.013	0.208 ± 0.009	0.540 ± 0.003
	1000	-0.553 ± 0.135	0.274 ± 0.011	0.087 ± 0.005	-0.085 ± 0.027	0.201 ± 0.005	0.536 ± 0.008
J-NPE	50	-2.708 ± 0.853	0.711 ± 0.157	0.067 ± 0.019	0.014 ± 0.045	0.619 ± 0.161	0.714 ± 0.053
	100	-1.995 ± 0.768	0.646 ± 0.213	0.076 ± 0.058	0.023 ± 0.045	0.550 ± 0.207	0.672 ± 0.068
	200	-1.730 ± 0.741	0.510 ± 0.226	0.064 ± 0.037	0.041 ± 0.056	0.406 ± 0.224	0.610 ± 0.106
	500	-0.362 ± 1.008	0.310 ± 0.027	0.078 ± 0.041	0.065 ± 0.050	0.203 ± 0.024	0.547 ± 0.012
	1000	0.560 ± 0.191	0.281 ± 0.015	0.064 ± 0.012	0.054 ± 0.012	0.169 ± 0.011	0.528 ± 0.005
FT	50	-3.225 ± 0.492	0.466 ± 0.041	0.152 ± 0.026	0.112 ± 0.010	0.401 ± 0.042	0.594 ± 0.018
	100	-2.480 ± 0.525	0.348 ± 0.016	0.195 ± 0.040	0.142 ± 0.032	0.281 ± 0.015	0.564 ± 0.008
	200	-0.804 ± 0.283	0.340 ± 0.020	0.064 ± 0.031	0.028 ± 0.055	0.246 ± 0.026	0.549 ± 0.004
	500	0.276 ± 0.132	0.271 ± 0.006	0.074 ± 0.027	0.061 ± 0.022	0.173 ± 0.009	0.533 ± 0.012
	1000	1.063 ± 0.053	0.242 ± 0.005	0.053 ± 0.014	0.046 ± 0.010	0.143 ± 0.004	0.519 ± 0.004

Table 4: Pendulum results. Mean ± sample standard deviation over three calibration seeds.

Method	n	LPP ↑	NRMSE ↓	Coverage MAE ↓	ACAUC (0 ideal)	Energy ↓	jC2ST (0.5 ideal)
Prior	–	-16.385	1.002	0.007	0.001	1.350	0.887
NPE_{base}	–	-54.382	1.039	0.477	0.311	1.650	0.967
SBI (oracle)	–	-7.491	0.436	0.010	0.007	0.478	0.521
PACE†	50	-16.964 ± 0.399	0.825 ± 0.011	0.063 ± 0.028	0.046 ± 0.013	1.081 ± 0.013	0.549 ± 0.019
	100	-15.507 ± 0.508	0.698 ± 0.029	0.066 ± 0.052	0.056 ± 0.036	0.896 ± 0.042	0.523 ± 0.014
	200	-14.305 ± 0.383	0.620 ± 0.042	0.056 ± 0.014	0.048 ± 0.013	0.783 ± 0.059	0.521 ± 0.011
	500	-12.303 ± 0.101	0.511 ± 0.011	0.048 ± 0.003	0.042 ± 0.003	0.613 ± 0.020	0.506 ± 0.002
	1000	-11.244 ± 0.039	0.482 ± 0.003	0.032 ± 0.004	0.033 ± 0.006	0.564 ± 0.004	0.504 ± 0.006
RoPE	50	-21.717 ± 0.108	0.614 ± 0.016	0.059 ± 0.003	-0.067 ± 0.003	0.765 ± 0.024	0.729 ± 0.007
	100	-21.457 ± 0.028	0.589 ± 0.004	0.067 ± 0.001	-0.078 ± 0.002	0.720 ± 0.007	0.705 ± 0.009
	200	-21.374 ± 0.028	0.586 ± 0.002	0.070 ± 0.001	-0.083 ± 0.000	0.711 ± 0.003	0.692 ± 0.004
	500	-21.309 ± 0.014	0.584 ± 0.002	0.070 ± 0.001	-0.083 ± 0.001	0.705 ± 0.002	0.686 ± 0.010
	1000	-21.286 ± 0.007	0.582 ± 0.002	0.071 ± 0.001	-0.085 ± 0.001	0.702 ± 0.002	0.688 ± 0.008
FRISBI	50	-15.839 ± 1.023	0.703 ± 0.047	0.032 ± 0.011	0.022 ± 0.018	0.891 ± 0.073	0.806 ± 0.025
	100	-14.062 ± 0.254	0.633 ± 0.007	0.021 ± 0.001	-0.013 ± 0.005	0.774 ± 0.014	0.765 ± 0.009
	200	-13.599 ± 0.105	0.619 ± 0.007	0.031 ± 0.005	-0.028 ± 0.001	0.745 ± 0.007	0.736 ± 0.006
	500	-13.099 ± 0.140	0.602 ± 0.006	0.035 ± 0.003	-0.038 ± 0.004	0.714 ± 0.011	0.714 ± 0.009
	1000	-12.911 ± 0.072	0.594 ± 0.002	0.041 ± 0.003	-0.045 ± 0.003	0.701 ± 0.004	0.706 ± 0.005
J-NPE	50	-16.018 ± 0.609	0.825 ± 0.102	0.064 ± 0.039	0.040 ± 0.047	1.089 ± 0.146	0.837 ± 0.042
	100	-15.229 ± 0.333	0.727 ± 0.068	0.068 ± 0.032	0.048 ± 0.027	0.945 ± 0.095	0.802 ± 0.031
	200	-15.294 ± 1.310	0.800 ± 0.182	0.034 ± 0.021	0.014 ± 0.016	1.046 ± 0.274	0.819 ± 0.078
	500	-13.754 ± 0.113	0.682 ± 0.033	0.038 ± 0.013	-0.011 ± 0.033	0.866 ± 0.044	0.780 ± 0.004
	1000	-13.866 ± 2.440	0.690 ± 0.273	0.021 ± 0.010	0.005 ± 0.016	0.877 ± 0.414	0.734 ± 0.142
FT	50	-16.852 ± 0.550	0.924 ± 0.101	0.056 ± 0.009	0.029 ± 0.012	1.234 ± 0.145	0.869 ± 0.036
	100	-16.788 ± 0.535	0.939 ± 0.091	0.062 ± 0.012	0.044 ± 0.007	1.250 ± 0.137	0.867 ± 0.040
	200	-14.409 ± 0.378	0.694 ± 0.108	0.045 ± 0.018	0.032 ± 0.020	0.896 ± 0.153	0.766 ± 0.037
	500	-14.216 ± 0.937	0.791 ± 0.089	0.032 ± 0.023	0.026 ± 0.023	1.015 ± 0.141	0.811 ± 0.045
	1000	-14.032 ± 2.391	0.719 ± 0.258	0.034 ± 0.001	0.019 ± 0.024	0.923 ± 0.389	0.761 ± 0.143

Table 5: Light Tunnel results. Mean ± sample standard deviation over three calibration seeds.

Method	n	LPP ↑	NRMSE ↓	Coverage MAE ↓	ACAUC (0 ideal)	Energy ↓	jC2ST (0.5 ideal)
Prior	–	-3.807	1.011	0.005	0.011	0.583	0.760
NPE_{base}	–	-15.227	0.803	0.785	0.499	0.739	0.973
SBI (oracle)	–	-2.152	0.176	0.016	0.012	0.094	0.506
PACE	50	-2.659 ± 0.098	0.196 ± 0.005	0.155 ± 0.002	0.128 ± 0.016	0.111 ± 0.004	0.544 ± 0.018
	100	-2.383 ± 0.011	0.186 ± 0.002	0.085 ± 0.024	0.064 ± 0.020	0.103 ± 0.001	0.515 ± 0.007
	200	-2.349 ± 0.009	0.189 ± 0.002	0.076 ± 0.014	0.062 ± 0.012	0.103 ± 0.001	0.515 ± 0.005
	500	-2.222 ± 0.015	0.179 ± 0.003	0.042 ± 0.007	0.032 ± 0.010	0.097 ± 0.001	0.505 ± 0.008
	1000	-2.172 ± 0.010	0.174 ± 0.001	0.031 ± 0.007	0.027 ± 0.003	0.094 ± 0.001	0.505 ± 0.016
RoPE	50	-2.388 ± 0.071	0.223 ± 0.010	0.068 ± 0.026	0.055 ± 0.028	0.118 ± 0.007	0.564 ± 0.012
	100	-2.352 ± 0.050	0.217 ± 0.011	0.068 ± 0.010	0.050 ± 0.008	0.114 ± 0.006	0.557 ± 0.011
	200	-2.318 ± 0.010	0.209 ± 0.005	0.062 ± 0.007	0.046 ± 0.004	0.110 ± 0.001	0.562 ± 0.007
	500	-2.263 ± 0.015	0.200 ± 0.002	0.049 ± 0.007	0.038 ± 0.007	0.105 ± 0.001	0.557 ± 0.015
	1000	-2.250 ± 0.006	0.196 ± 0.001	0.050 ± 0.005	0.040 ± 0.003	0.103 ± 0.000	0.553 ± 0.010
FRISBI	50	-2.696 ± 0.147	0.203 ± 0.010	0.220 ± 0.027	0.167 ± 0.014	0.117 ± 0.007	0.545 ± 0.021
	100	-2.515 ± 0.137	0.200 ± 0.020	0.095 ± 0.031	0.082 ± 0.026	0.111 ± 0.011	0.528 ± 0.005
	200	-2.511 ± 0.090	0.190 ± 0.011	0.184 ± 0.028	0.143 ± 0.025	0.107 ± 0.007	0.507 ± 0.006
	500	-2.379 ± 0.014	0.179 ± 0.002	0.152 ± 0.018	0.113 ± 0.016	0.099 ± 0.001	0.507 ± 0.003
	1000	-2.286 ± 0.011	0.169 ± 0.002	0.138 ± 0.001	0.108 ± 0.003	0.093 ± 0.001	0.498 ± 0.004
J-NPE	50	-3.172 ± 0.045	0.257 ± 0.013	0.229 ± 0.033	0.180 ± 0.036	0.148 ± 0.003	0.572 ± 0.010
	100	-2.762 ± 0.090	0.252 ± 0.015	0.146 ± 0.018	0.133 ± 0.022	0.137 ± 0.009	0.557 ± 0.019
	200	-2.777 ± 0.184	0.244 ± 0.008	0.172 ± 0.085	0.133 ± 0.059	0.133 ± 0.004	0.554 ± 0.010
	500	-2.594 ± 0.064	0.228 ± 0.003	0.134 ± 0.056	0.116 ± 0.051	0.125 ± 0.001	0.533 ± 0.022
	1000	-2.394 ± 0.025	0.201 ± 0.007	0.093 ± 0.011	0.073 ± 0.009	0.107 ± 0.002	0.513 ± 0.019
FT	50	-2.805 ± 0.213	0.211 ± 0.006	0.235 ± 0.041	0.176 ± 0.023	0.122 ± 0.007	0.561 ± 0.018
	100	-2.585 ± 0.129	0.205 ± 0.012	0.168 ± 0.036	0.124 ± 0.020	0.114 ± 0.007	0.528 ± 0.011
	200	-2.511 ± 0.177	0.195 ± 0.006	0.169 ± 0.070	0.128 ± 0.052	0.109 ± 0.006	0.513 ± 0.001
	500	-2.251 ± 0.014	0.181 ± 0.000	0.061 ± 0.005	0.052 ± 0.003	0.098 ± 0.001	0.511 ± 0.004
	1000	-2.171 ± 0.002	0.170 ± 0.001	0.057 ± 0.013	0.051 ± 0.009	0.092 ± 0.000	0.504 ± 0.012

Table 6: Wind Tunnel results. Mean ± sample standard deviation over three calibration seeds.

C.1 PARAMETER EFFICIENT ADAPTATION

Task	Full NPE # params	PACE adapter # params	Trainable fraction
CS	33,602	1,044	3.11%
SIR	28,088	896	3.19%
Pendulum	1,474,964	6,264	0.42%
Light Tunnel	476,963	5,774	1.21%
Wind Tunnel	170,929	1,678	0.98%

Table 7: Parameter counts for each pretrained NPE and its PACE adapters.

Table 7 shows that PACE trains only 0.42% to 3.19% as many parameters as the full NPE. This corresponds to between 31 and 235 times fewer trainable parameters across the five tasks. The reduction is especially large for Pendulum, where 6264 affine parameters adapt an NPE with nearly 1.5 million parameters. Since the pretrained weights remain frozen, each target domain requires storing and optimizing only its small set of scales and shifts. PACE therefore obtains its competitive performance with a substantially smaller adaptation model and without retraining the full posterior estimator.

CS task The NPE encoder first standardizes the four summary statistics using moments computed from the simulated training set. It then applies a multilayer perceptron with widths 4, 12, 48, 48, 36, and 9. ReLU activations follow every hidden layer and the resulting nine dimensional vector conditions the flow. The three dimensional posterior uses one spline transform with three hidden conditioner layers of width 100. Parameters are standardized using their simulation prior moments. The NPE is trained on 50000 simulations and validated on 100000 simulations with learning rate 5×10^{-4}, batch size 100, and at most 10^6 updates. Validation is performed every 1000 updates with patience 10. Training terminated after 26000 updates. PACE adapts nine learned layers with 1044 trainable parameters. It uses learning rate 10^{-4}, batch size 100, at most 10000 updates, validation every 100 updates, and patience 50. Results are averaged over calibration seeds 0, 1, and 2.

SIR task The six summary statistics are standardized using the simulated training set. The encoder is a multilayer perceptron with widths 6, 8, 32, 32, 24, and 6, with ReLU activations after all hidden layers. Its six dimensional output conditions a two dimensional spline flow with one transform and three hidden conditioner layers of width 100. Parameter standardization uses the moments of the triangular simulation prior. The NPE uses 50000 training simulations and 100000 validation simulations. We train it with learning rate 5×10^{-4}, batch size 100, at most 10^6 updates, validation every 1000 updates, and patience 10. Training terminated after 42000 updates. PACE adapts nine learned layers with 896 trainable parameters. It uses learning rate 10^{-4}, batch size 100, at most 10000 updates, validation every 100 updates, and patience 50. Results are averaged over calibration seeds 0, 1, and 2.

Pendulum task The encoder receives a length 200 trajectory and uses seven one dimensional convolutions with channel widths 16, 64, 128, 128, 128, 128, and 128. The first convolution has stride 1. The second, fourth, and sixth convolutions have stride 2. All convolutions use kernel size 3 and dilation 2. Average pooling follows the second, fourth, and sixth convolutions. The flattened features pass through linear layers with widths 512, 128, 32, and 10, with ReLU activations before the final context. The two dimensional posterior uses five spline transforms. Each conditioner has three hidden layers of width 100. Parameters are standardized using the moments of the uniform simulation prior. The NPE is trained on 50000 simulations and validated on 10000 simulations using learning rate 10^{-4}, batch size 100, at most 100 epochs, and patience 10. Validation is performed after every epoch and training terminated after epoch 43. PACE adapts 31 learned layers with 6264 trainable parameters. It uses learning rate 10^{-2}, batch size 8, at most 10000 updates, validation every 100 updates, and patience 50. Results are averaged over calibration seeds 0, 1, and 2.

Light Tunnel task The NPE processes a three channel image of size 100×100. Its encoder uses five two dimensional convolutions with output channels 64, 128, 128, 64, and 3. The first three convolutions use kernel size 3 and stride 2. Max pooling follows the second convolution. The final two convolutions use kernel size 1. After flattening, linear layers of widths 100 and 20 produce the

flow context. ReLU activations follow every convolution and the first linear layer. RGB parameters are divided by 255 and the polarization parameter is left unchanged. The four dimensional posterior uses five spline transforms with three hidden conditioner layers of width 128. The NPE is trained on 20000 rendered images and validated on 10000 rendered images using learning rate 5×10^{-4}, batch size 100, at most 1000 epochs, and patience 50. Validation is performed after every epoch and the selected checkpoint is from epoch 165. The reported PACE runs adapt 27 encoder and flow layers with 5774 trainable parameters. These runs use the adapted final encoder output as the flow context and precede the separate additional context map in the current implementation. PACE uses learning rate 10^{-3} and trains for at most 300 epochs with validation after every epoch, patience 100, and gradient norm clipping at 10. The batch size is 8 for 50 calibration pairs and 16 for all other sizes. Results are averaged over calibration seeds 42, 43, and 44.

Wind Tunnel task Each length 50 pressure trace is standardized at every time point using the simulated training set. The encoder uses one dimensional convolutions with output channels 16, 64, 128, and 128. The second and fourth convolutions use stride 2. Average pooling follows the second convolution and adaptive average pooling reduces the final sequence to length 4. Linear layers with widths 128, 32, and 10 then produce the context. ReLU activations follow every hidden layer. The scalar posterior uses one spline transform with three hidden conditioner layers of width 100. The NPE is trained on 50000 simulations and validated on 10000 simulations using learning rate 10^{-4}, batch size 256, at most 150 epochs, and patience 15. Validation is performed after every epoch and training stopped after 101 epochs. PACE adapts 12 encoder, context, and flow layers with 1678 trainable parameters. It uses learning rate 10^{-3}, at most 10000 updates, validation every 100 updates, patience 50, and gradient norm clipping at 10. The batch size is 8 for 50 calibration pairs and 16 for all other sizes. Results are averaged over calibration seeds 0, 1, and 2.

D Theoretical Analysis of PACE

In this section, we provide a compact sufficient-condition result that formalizes when lightweight feature-wise calibration of PACE can recover the exact target posterior. Note that we do not claim that arbitrary simulator misspecification is affine. Rather, the result characterizes a realizable regime that is explicitly targeted by PACE.

D.1 Setup and induced representation distributions

Let

$$x_s \sim p_s(x \mid \theta), \qquad x_o \sim p_o(x \mid \theta),$$

denote simulator and target observations, respectively, and let the frozen simulator-trained encoder be

$$h_\psi : \mathcal{X} \to \mathcal{Z}.$$

For the representation-calibration subfamily of PACE, define

$$z_s = h_\psi(x_s), \qquad z_o^C = T_C(h_\psi(x_o)), \qquad T_C(z) = \gamma \odot z + \beta.$$

Rather than introducing $p(z \mid \theta)$ as a primitive model distribution, we define the corresponding *induced* conditional distributions:

$$p_s^z(\cdot \mid \theta) := (h_\psi)_{\#} p_s(\cdot \mid \theta),$$

and

$$p_o^{z,C}(\cdot \mid \theta) := (T_C \circ h_\psi)_{\#} p_o(\cdot \mid \theta),$$

where $\#$ denotes the pushforward of a distribution. Equivalently, p_s^z and $p_o^{z,C}$ are simply the distributions obtained by sampling an observation from the corresponding domain and passing it through the stated deterministic representation map.

The simulator-pretrained posterior estimator is written as

$$q_\phi(\theta \mid z).$$

Relation to full PACE. PACE applies feature-wise scale-and-shift maps at several locations along the frozen inference pathway, including the encoder, conditioning interface, and flow. The result below analyzes the simpler representation-calibration subfamily obtained by correcting the representation supplied to the pretrained posterior estimator. Since this subfamily is contained in the full PACE adaptation family, the result provides a sufficient condition under which full PACE also contains the exact target posterior.

D.2 Exact recovery under affine representational misspecification

Proposition 1 (Exact recovery under affine representational misspecification)**.** *Assume that the simulator and target domains share the same parameter prior $p(\theta)$. Suppose there exists a calibration parameter $C^\star$ such that:*

1. ***Target sufficiency.*** *With $z_o^\star := T_{C^\star}(h_\psi(x_o))$, we have $\theta \perp x_o \mid z_o^\star$. Equivalently, $p_o(\theta \mid x_o) = p_o(\theta \mid z_o^\star)$.*
2. ***Conditional alignment of induced representations.*** *For every θ, $p_o^{z,C^\star}(\cdot \mid \theta) = p_s^z(\cdot \mid \theta)$.*
3. ***Exact simulator posterior estimation in representation space.*** *For representations in the support of p_s^z, $q_\phi(\theta \mid z) = p_s(\theta \mid z)$.*

Then, for target observations,

$$q_\phi(\theta \mid T_{C^\star}(h_\psi(x_o))) = p_o(\theta \mid x_o).$$

Proof. Let $z_o^\star = T_{C^\star}(h_\psi(x_o))$. By Assumption 2, the target and simulator domains induce the same conditional distribution in representation space: $p_o(z \mid \theta) = p_s(z \mid \theta)$, where this equality is shorthand for equality of the induced measures $p_o^{z,C^\star}(\cdot \mid \theta)$ and $p_s^z(\cdot \mid \theta)$.

Because the parameter prior is also shared, Bayes' rule implies equality of the corresponding posteriors in representation space: $p_o(\theta \mid z) = p_s(\theta \mid z)$. By Assumption 3, $p_s(\theta \mid z) = q_\phi(\theta \mid z)$. Evaluating at $z = z_o^\star$ gives $p_o(\theta \mid z_o^\star) = q_\phi(\theta \mid z_o^\star)$. Finally, Assumption 1 gives $p_o(\theta \mid x_o) = p_o(\theta \mid z_o^\star)$. Combining the equalities yields

$$q_\phi(\theta \mid T_{C^\star}(h_\psi(x_o))) = p_o(\theta \mid x_o).$$

□

Interpretation. Proposition 1 is a realizability statement. It says that if the frozen simulator-trained representation already preserves all target-domain information needed for inference, and the remaining simulator-to-target discrepancy can be removed within the affine calibration family, then updating the pretrained encoder or posterior estimator is unnecessary for exact recovery.

The proposition does *not* state that arbitrary misspecification is affine or that PACE is universally sufficient. When the target representation cannot be aligned to the simulator representation within this restricted family, PACE incurs an approximation gap, motivating the comparison with more expressive adaptation methods such as full fine-tuning.

D.3 Approximation–estimation perspective

Let us denote by $\mathcal{Q}_{\mathrm{PACE}}$ the target posterior family reachable by PACE and $\mathcal{Q}_{\mathrm{FT}}$ the larger family reachable, assuming *unrestricted* fine-tuning. Since PACE freezes the pretrained parameters and optimizes only a restricted calibration family,

$$\mathcal{Q}_{\mathrm{PACE}} \subseteq \mathcal{Q}_{\mathrm{FT}}.$$

Let us define the population target negative log-likelihood as

$$R(q) = \mathbb{E}_{(\theta,x)\sim p_o}\left[-\log q(\theta \mid x)\right].$$

For an empirical PACE solution $\widehat{q}_{\mathrm{PACE}}$, the excess risk can be decomposed as

$$R(\widehat{q}_{\mathrm{PACE}}) - R(q_o^\star) = \underbrace{\inf_{q\in\mathcal{Q}_{\mathrm{PACE}}} R(q) - R(q_o^\star)}_{\text{approximation error}} + \underbrace{R(\widehat{q}_{\mathrm{PACE}}) - \inf_{q\in\mathcal{Q}_{\mathrm{PACE}}} R(q)}_{\text{estimation error}},$$

where $q_o^\star$ denotes the optimal target posterior in the unrestricted population limit.

Thie above decomposition highlights an intended tradeoff. Restricting adaptation can increase approximation error relative to full fine-tuning, but can reduce finite-sample estimation error when the labeled target calibration set is small. Under standard boundedness and regularity assumptions, a generic uniform convergence argument yields an estimation term scaling schematically as

$$O\left(\sqrt{\frac{\mathcal{C}_{\mathrm{PACE}} + \log(1/\delta)}{N}}\right),$$

where $\mathcal{C}_{\mathrm{PACE}}$ is an appropriate complexity measure of the PACE adaptation family and N is the number of labeled target samples.

Practical implication of the analysis. This perspective also offers us useful guideline/prescription for a method like PACE. In particular, it shows predicts that PACE should be most competitive when (i) simulator pretraining preserves the information needed for target inference, (ii) the simulator-to-target discrepancy can be approximated by a low-complexity calibration, and (iii) the labeled target set is small. As the representation mismatch grows, the approximation error of the restricted PACE family may dominate, at which point a more expressive method such as full fine-tuning can become advantageous.