LazyDINO: Fast, Scalable, and Efficiently Amortized Bayesian Inversion via Structure-Exploiting and Surrogate-Driven Measure Transport

Lianghao Cao[b,*], Joshua Chen[c,*], Michael Brennan[a], Thomas O'Leary-Roseberry[c], Youssef Marzouk[a], Omar Ghattas[c,d]

[a] *Center for Computational Science and Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA*
[b] *Department of Computing and Mathematical Sciences, California Institute of Technology Pasadena, CA 91125, USA*
[c] *Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, 201 E. 24th Street, C0200, Austin, TX 78712, USA*
[d] *Walker Department of Mechanical Engineering, The University of Texas at Austin, 204 E. Dean Keaton Street, Austin, TX 78712, USA*

Abstract

We present `LazyDINO`, a transport map variational inference method for fast, scalable, and efficiently amortized solutions of high-dimensional nonlinear Bayesian inverse problems with expensive parameter-to-observable (PtO) maps. Our method consists of an offline phase in which we construct a derivative-informed neural surrogate of the PtO map using joint samples of the PtO map and its Jacobian. During the online phase, when given observational data, we seek rapid posterior approximation using surrogate-driven training of a lazy map [Brennan et al., *NeurIPS*, (2020)], i.e., a structure-exploiting transport map with low-dimensional nonlinearity. The trained lazy map then produces approximate posterior samples or density evaluations. Our surrogate construction is optimized for amortized Bayesian inversion using lazy map variational inference. We show that (i) the derivative-based reduced basis architecture [O'Leary-Roseberry et al., *Comput. Methods Appl. Mech. Eng.*, 388 (2022)] minimizes the upper bound on the expected error in surrogate posterior approximation, and (ii) the derivative-informed training formulation [O'Leary-Roseberry et al., *J. Comput. Phys.*, 496 (2024)] minimizes the expected error due to surrogate-driven transport map optimization. Our numerical results demonstrate that `LazyDINO` is highly efficient in cost amortization for Bayesian inversion. We observe one to two orders of magnitude reduction of offline cost for accurate posterior approximation, compared to simulation-based amortized inference via conditional transport and conventional surrogate-driven transport. In particular, `LazyDINO` outperforms Laplace approximation consistently using fewer than 1000 offline samples, while other amortized inference methods struggle and sometimes fail at 16,000 offline samples.

Keywords: Bayesian inverse problem, variational inference, measure transport, surrogate model, dimension reduction, derivative-informed operator learning

1. Introduction

We investigate the solution of nonlinear *Bayesian inverse problems* (BIPs), i.e., inferring uncertain parameters of computational models from sparse, noisy, and indirect observational data. Let $m \in \mathscr{M}$ denote the unknown model parameter and assume the observational data vector $\boldsymbol{y} \in \mathbb{R}^{d_{\boldsymbol{y}}}$ is given by:

$$\boldsymbol{y} = \boldsymbol{\mathcal{G}}(m) + \boldsymbol{n}, \quad \boldsymbol{n} \sim \mathcal{N}(0, \Gamma_n),$$

*Co-first author, both are corresponding authors.
Email addresses: `lianghao@caltech.edu` (Lianghao Cao), `joshuawchen@utexas.edu` (Joshua Chen), `mcbrenn@mit.edu` (Michael Brennan), `tom.olearyroseberry@utexas.edu` (Thomas O'Leary-Roseberry), `ymarz@mit.edu` (Youssef Marzouk), `omar@oden.utexas.edu` (Omar Ghattas)

where $\boldsymbol{\mathcal{G}} : \mathscr{M} \to \mathbb{R}^{d_{\boldsymbol{y}}}$ is the parameter-to-observable (PtO) map and $\boldsymbol{n} \in \mathbb{R}^{d_{\boldsymbol{y}}}$ is an unknown noise vector. Given a parameter prior distribution μ, we seek to characterize the posterior distribution $\mu^{\boldsymbol{y}}$ defined via Bayes' rule

$$\underbrace{\mathrm{d}\mu^{\boldsymbol{y}}(m)}_{\text{Posterior}} \propto \underbrace{\exp\left(-\frac{1}{2}\left\|\Gamma_n^{-1/2}\left(\boldsymbol{\mathcal{G}}(m)-\boldsymbol{y}\right)\right\|^2\right)}_{\text{Likelihood}} \underbrace{\mathrm{d}\mu(m)}_{\text{Prior}}.$$

We are particularly interested in continuum models for physical systems, e.g., parametric partial differential equations (PDEs), where the parameter $m \in \mathscr{M}$ can have arbitrarily high dimensions, such as spatially varying parameter fields, and the PtO map $\boldsymbol{\mathcal{G}}$ is defined implicitly through the solution of the governing equations [1–4]. This type of BIP is challenging due to (i) the high computational cost of likelihood evaluations due to model solutions, (ii) the difficulty of characterizing high-dimensional posterior distributions due to the curse of dimensionality, and (iii) non-Gaussianity of the posterior distribution. These challenges are acute limitations when one seeks fast solutions of BIPs for a range of observational data, as in real-time uncertainty quantification for predictive digital twins [5] and optimal experimental design [6]. Solving BIPs in this setting requires methods with *amortized computational cost*—that is, most of the expensive computation is performed offline, i.e., before acquiring the data, and posterior characterization incurs a comparatively negligible cost once the data is available. These challenges demand methodological innovations beyond conventional approaches such as Markov chain Monte Carlo (MCMC). In this work, we integrate recent advances in dimension reduction, neural operator learning, and measure transport to derive a fast, scalable, and efficiently amortized method for BIPs that is well-suited to modern computing frameworks.

1.1. Variational inference using lazy maps

We consider using transport map variational inference (TMVI) to approximate the posterior $\mu^{\boldsymbol{y}}$. This method seeks to construct a parameterized transport map $\mathcal{T}_{\boldsymbol{\theta}} : \mathscr{M} \to \mathscr{M}$ between a reference distribution, which we take to be the prior μ, and the target Bayesian posterior $\mu^{\boldsymbol{y}}$. The map parameters can be found by minimizing the reverse KL divergence (rKL):

$$\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}(\mathcal{T}_{\boldsymbol{\theta}\sharp}\mu \| \mu^{\boldsymbol{y}}), \tag{1}$$

where $(\cdot)_\sharp$ denotes the pushforward of a probability distribution. Once the transport map is constructed, it allows for fast on-demand approximate posterior sampling through map evaluations $\mathcal{T}_{\boldsymbol{\theta}}(m^{(j)})$ of the reference samples $m^{(j)} \overset{\text{i.i.d.}}{\sim} \mu$. However, it can be difficult to represent expressive transport maps in high dimensions. For example, triangular maps [7] on $\mathbb{R}^n$ must describe n-variate functions and thus suffer from the curse of dimensionality. Kernel-based methods, such as Stein variational inference [8–10], lose expressiveness in high dimensions. Flow-based methods [11, 12] often increase expressiveness by adding layers, which is typically performed ad hoc and require tuning.

Lazy maps [13] are a class of transport maps that alleviate the curse of dimensionality by restricting the nonlinearity of the map to a relatively low-dimensional parameter latent space. Let $\mathcal{E}_r : \mathscr{M} \to \mathbb{R}^{d_r}$ with $d_r \ll \dim(\mathscr{M})$ be a linear encoder such that $\mathrm{Im}(\mathcal{E}_r) = \mathbb{R}^{d_r}$ defines the parameter latent space. A lazy map has the following form:

$$\mathcal{T}_{\boldsymbol{\theta}} \coloneqq (\mathrm{Id}_{\mathscr{M}} - \mathcal{E}_r \circ \mathcal{D}_r) + \mathcal{D}_r \circ \mathbf{T}_{\boldsymbol{\theta}} \circ \mathcal{E}_r, \tag{2}$$

where Id denotes the identity map, $\mathcal{D}_r : \mathbb{R}^{d_r} \to \mathscr{M}$ is a linear decoder, and $\mathbf{T}_{\boldsymbol{\theta}} : \mathbb{R}^{d_r} \to \mathbb{R}^{d_r}$ is a parametrized latent space transport map. The lazy map approximates the posterior in the latent space using TMVI while the prior fills the complementary space of $\mathrm{Im}(\mathcal{D}_r)$. When the prior is Gaussian, and the encoder is a whitening transformation, the rKL minimization problem becomes

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{m\sim\mu}\left[\frac{1}{2}\left\|\Gamma_n^{-1/2}\left(\left(\boldsymbol{\mathcal{G}} \circ \mathcal{T}_{\boldsymbol{\theta}}\right)(m)-\boldsymbol{y}\right)\right\|^2 + \frac{1}{2}\left\|\mathbf{T}_{\boldsymbol{\theta}}(\mathcal{E}_r m)\right\|^2 - \log\left|\det \nabla \mathbf{T}_{\boldsymbol{\theta}}(\mathcal{E}_r m)\right|\right]. \tag{3}$$

A key component of lazy maps is finding a parameter subspace that captures the discrepancy between the prior and posterior distribution. This parameter subspace is often known as the likelihood-informed

subspace [14] or active subspace [15], which is known to exist for a large class of inverse problems and can be found via solving eigenvalue problems based on score functions; see, e.g., [16–23]. By exploiting the structure of the BIPs, TMVI using lazy maps typically achieves high-quality posterior approximation more efficiently than TMVI without parameter reduction or with alternative reduction techniques.

Another fundamental challenge of TMVI, including when a lazy map is used, lies in the high cost of transport map training, which requires solving the stochastic and model-constrained rKL minimization problem in (3). Numerous evaluations of the PtO map $\boldsymbol{\mathcal{G}}$ and the actions of its Jacobian $D\boldsymbol{\mathcal{G}}$ are required within each optimization iteration. These evaluations involve repeated solutions of the governing equations of the computational models and their forward or adjoint sensitivities, which can be prohibitively expensive when these equations are, e.g., large-scale nonlinear PDEs. This cost barrier becomes further exacerbated when multiple posteriors need to be approximated for different instances of observational data.

1.2. Derivative-informed surrogate for amortized lazy map variational inference

In this work, we remove the computational bottleneck of model solutions in lazy map training by constructing a fast-to-evaluate ridge function surrogate of the PtO map using a neural network latent representation $\mathbf{g}_{\boldsymbol{w}} : \mathbb{R}^{d_r} \to \mathbb{R}^{d_{\boldsymbol{y}}}$:

$$\boldsymbol{\mathcal{G}}(m) \approx \boldsymbol{V}\mathbf{g}_{\boldsymbol{w}}(\mathcal{E}_r m),$$

where $\boldsymbol{w}$ is the weight of the neural network and $\boldsymbol{V}$ is a (reduced) basis on the data space. This surrogate architecture that uses the same parameter reduction technique as lazy maps is the derivative-informed neural network (`DIPNet`) in [24], which belongs to a larger class of reduced basis neural operator [25, 26]. Once the surrogate is constructed, we perform TMVI in the parameter latent space using the surrogate-driven rKL objective for the given observational data $\boldsymbol{y}$:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,\mathrm{Id}_{\mathbb{R}^{d_r}})}\left[\frac{1}{2}\left\|(\mathbf{g}_{\boldsymbol{w}}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - \boldsymbol{V}^*\boldsymbol{y}\right\|^2 + \frac{1}{2}\left\|\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})\right\|^2 - \log\left|\det\nabla\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})\right|\right].$$

The forward and Jacobian evaluation costs of the neural network $\mathbf{g}_{\boldsymbol{w}}$ are significantly lower than those of model and sensitivity solutions. As a result, the high cost of the PtO map and its Jacobian evaluations for optimizing lazy maps are amortized.

Conventionally, the neural network is trained offline using samples of the PtO map via mean squared error minimization [24, 27]:

$$\min_{\boldsymbol{w}} \mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{V}^*\boldsymbol{\mathcal{G}}(m) - \mathbf{g}_{\boldsymbol{w}}(\mathcal{E}_r m)\right\|^2\right].$$

When a limited number of PtO map samples is used to estimate the expectation, the trained surrogate may be inadequate for lazy map training as the surrogate Jacobian error is not directly controlled. This, in turn, leads to inaccurate gradients of the rKL objective and substantial gaps between the objective values at the true optimum and the surrogate-approximated optimum (which we refer to herein as optimality gaps). Similarly, conventionally trained neural operator surrogates struggle to accelerate other gradient-based optimization in high or infinite dimensions; see, e.g., optimal design [28] and geometric MCMC [29].

In this work, we follow [29, 30] and train `DIPNet` PtO map surrogates using a derivative-informed learning method, which exerts surrogate error control in the Sobolev space of the PtO map latent representation using joint samples of the PtO map $\boldsymbol{\mathcal{G}}$ and its Jacobian $D\boldsymbol{\mathcal{G}}$:

$$\min_{\boldsymbol{w}} \mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{V}^*\boldsymbol{\mathcal{G}}(m) - \mathbf{g}_{\boldsymbol{w}}(\mathcal{E}_r m)\right\|^2 + \left\|\boldsymbol{V}^* D\boldsymbol{\mathcal{G}}(m)\mathcal{D}_r - \nabla\mathbf{g}_{\boldsymbol{w}}(\mathcal{E}_r m)\right\|_F^2\right].$$

We show that derivative-informed learning of `DIPNet` surrogate is equivalent to minimizing an upper bound on posterior approximation error as well as the optimality gap of surrogate-driven lazy map training for amortized Bayesian inversion. We refer to this surrogate construction with optimized architecture and training as reduced basis derivative-informed neural operator (`RB-DINO`) [30].

1.3. Solving high-dimensional nonlinear Bayesian inverse problems using `LazyDINO`

Combining TMVI using lazy maps and `RB-DINO` surrogate construction creates a competitive method for amortized solutions of high-dimensional model-constrained BIPs, referred to as `LazyDINO`. The method is composed of offline and online phases.

Offline phase. We first generate samples of the PtO map and its Jacobian by solving the governing equations of the computational model and its forward or adjoint sensitivity. These samples are then used to construct a `RB-DINO` surrogate of the PtO map.

Online phase. After collecting observational data, we seek rapid posterior approximation via `RB-DINO` surrogate-driven training of a latent space transport map. The trained transport map can be used to produce approximate posterior samples. This process can be repeated for different observational data, effectively amortizing the construction cost of the `RB-DINO` surrogate.

We provide extensive numerical studies that compare `LazyDINO` against a range of TMVI methods, including the Laplace approximation, simulation-based amortized inference (SBAI) via conditional transport, TMVI using lazy maps, and lazy maps combined with conventional surrogate construction (`LazyNO`); see Section 1.4.4 and Table 2. We test these methods on two BIPs, each with four different instances of observation data: (i) inferring the diffusivity field in a nonlinear reaction–diffusion PDE and (ii) inferring the heterogeneous material property of a hyperelastic material thin film. We devise extensive posterior approximation tests, including moment discrepancies, probability density-based metrics, and various posterior visualizations (e.g., marginals, mean, MAP estimates, and point-wise marginal variance).

The main contributions of the `LazyDINO` method are summarized below.

(**C**1) The `RB-DINO` surrogate construction is optimized for amortized lazy map variational inference.

- *Surrogate architecture.* In Theorem 3.1, we derive upper bounds for the expected posterior approximation error when a neural ridge function surrogate replaces the PtO map in the likelihood. This result, which is a straightforward extension from those in [29, 31], bounds the forward KL (fKL) averaged over the marginal observational data distribution by the sum of a parameter reduction error and a latent representation error. Minimizing this error upper bound gives rise to the `DIPNet` architecture [24], where the parameter encoder is found by derivative-informed dimension reduction [15, 32] using samples of the PtO map Jacobian.
- *Derivative-informed learning.* We show that the expected gradient error (Theorem 3.2), and the expected optimality gap (Corollary 3.3) in surrogate-driven lazy map optimization can be controlled by a weighted Sobolev norm of the surrogate approximation error. This error measure is consistent with the objective function in derivative-informed operator learning [29, 30] that uses joint samples of the PtO map and its Jacobian for surrogate training. In other words, derivative-informed learning of reduced basis neural networks (`RB-DINO`) minimizes the expected error in the stochastic optimization of lazy maps due to the surrogate representation.

(**C**2) `LazyDINO` enables fast, scalable, and efficiently amortized solutions of high-dimensional Bayesian inverse problems.

- *Scabalility.* The surrogate and transport map training in `LazyDINO` are independent of the parameter dimension as we co-design their latent representations in the same relatively low-dimensional parameter subspace that captures prior-to-posterior updates.
- *Fast online inference.* Using a neural network surrogate rKL objective for transport map training, `LazyDINO` circumvents the computational bottleneck of model solutions and fully exploits GPU-based accelerations to rapidly approximate posteriors. We demonstrate that the optimize-then-sample approach of `LazyDINO` leads to faster online sampling than the typical inversion-to-sample approach of SBAI.

- *Superior cost–accuracy trade-off in amortized Bayesian inversion.* The `RB-DINO` surrogate construction is highly efficient in cost amortization for solving BIPs, i.e., it achieves high posterior approximation error at low offline training cost. In our numerical example, we observed one to two orders of magnitude of cost reduction in offline computation for achieving similar accuracy in posterior approximation compared to `LazyNO` and SBAI. Moreover, `LazyDINO` consistently outperforms Laplace approximation at a small training sample regime ($< 1,000$). In contrast, `LazyNO` and SBAI struggle to outperform Laplace approximation and, in some cases, failed at $16,000$ training samples.

1.4. Related works

In the following subsections, we discuss related work in dimension reduction, surrogate modeling, and variational inference for BIPs.

1.4.1. Baseline: The Laplace Approximation

The Laplace approximation (LA) constructs a Gaussian approximation of the posterior, leading to efficient sampling and density evaluations [33]. This makes the LA a sensible baseline for settings that require fast approximate posterior sampling. The LA construction requires (1) MAP point estimation, followed by (2) covariance estimation via solving a generalized eigenvalue problem for the Hessian of the negative log-posterior at the MAP point. These Hessians often have low effective numerical rank, allowing for efficient implementations in practice [2, 4, 34]. Details on LA are included in Appendix G.

1.4.2. Dimension reduction for Bayesian inverse problems

A common likelihood-independent dimension reduction technique is the Karhunen–Loève expansion, which represents the parameter in a finite (small) number of prior covariance eigenfunctions, see e.g., see [35]. Derivative-based dimension reduction techniques identify the parameter subspace that the likelihood is most sensitive to, in prior or posterior expectation, and thereby provide more targeted dimension reduction and greater efficiency [14, 19, 32, 36, 37]. This subspace, often referred to as the likelihood-informed subspace, is related to the Fisher information and is shown to be optimal with respect to the KL divergence in [36]. In [31], the authors show that a subspace computed by averaging Fisher information over the prior distribution is optimal on average over the marginal distribution of observational data.

1.4.3. Surrogate models for Bayesian inverse problems

Substantial work has been done on using surrogate models, e.g., polynomial approximation [35, 38–40] and model-order reduction [41–43], to accelerate solutions of BIPs. Surrogate models are often used within multi-fidelity posterior sampling algorithms [44–46].

This work focuses on neural network surrogates due to their high flexibility, rich approximation properties, and scalability. Since our algorithmic framework can be applied to infinite-dimensional BIPs, we note the connection to neural operator surrogates [26] that map between function spaces with architectures and training agnostic to the discretization of these spaces. Notable architectures include reduced basis neural networks using linear [24, 25, 47–49] or nonlinear [50, 51] dimension reduction, and neural network integral kernels such as Fourier neural operator and its variants [52–55]. Notable training formulations include conventional supervised learning using input-output samples and physics-informed learning [56, 57] with additional loss functions related to the residual of the implicit equation (e.g., PDE residuals).

Our surrogate architecture is based on the aforementioned derivative-based dimension reduction strategy, i.e., `DIPNet` in [24, 49]. We advocate for the derivative-informed operator learning method for surrogate training, i.e., derivative-informed neural operator (DINO) [30]. This method has been successfully applied to surrogate-driven solutions of PDE-constrained optimization under uncertainties [28], Bayesian optimal experimental design [58, 59], and infinite-dimensional BIPs [29]. Recent work [60] also explored training the deep operator network (DeepONet) architecture using this method.

1.4.4. Transport map parameterizations and amortized inference

The `LazyDINO` algorithm performs TMVI for a reduced dimensional inference problem in a specifically chosen latent space. It assumes no particular map parameterization; rather it wraps around any provided transport map class. We briefly review popular transport map parameterizations and provide references for further reading. Normalizing flows (see [11, 12, 61, 62]) form a broad class of methods that construct transport maps through compositions of neural networks with specific parameterizations. Autoregressive flows [63–68], a popular subclass, compose autoregressive (triangular) maps to allow efficient computation of Jacobian determinants [69, 70]. Several works seek an approximation to the Knothe-Rosenblatt (KR) rearrangement [71, 72], a diffeomorphic triangular map that exists between any two distributions that are absolutely continuous with respect to a common measure, using orthonormal basis expansions (e.g., sparse polynomials) [73–78] or neural networks [79]. One distinguishing feature of `LazyDINO` is its use of PtO map Jacobian evaluations during training. Other recent works also incorporate derivative information, such as by adding a Fisher divergence term to the training objective [80, 81], thus exploiting the differentiability of the log-likelihood. Finally, we note the recent rise in inference methods that amortize transport-based posterior approximation [82–86]. Simulation-based amortized inference (SBAI) [87] approaches parameterize transport maps to treat the conditioning variable (i.e., the observation) as a functional input and generate samples from the corresponding posterior. We compare `LazyDINO` with SBAI in our numerical examples.

1.5. Notation

- We use bold symbols to denote finite-dimensional vectors, e.g., $\boldsymbol{x} \in \mathbb{R}^{d_{\boldsymbol{x}}}$ where $d_{\boldsymbol{x}}$ is the dimension. We denote the 2-norm on finite-dimensional vector spaces as $\|\cdot\|$. We denote the Frobenius matrix norm as $\|\cdot\|_F$. We use math script to denote separable Hilbert spaces that have high or infinite dimensions, e.g., $\mathscr{X}$, with inner product $\langle\cdot,\cdot\rangle_{\mathscr{X}}$, norm $\|\cdot\|_{\mathscr{X}}$ and element $x \in \mathscr{X}$.

- We denote the Banach space of bounded linear operators from $\mathscr{X}_1$ to $\mathscr{X}_2$ as $B(\mathscr{X}_1, \mathscr{X}_2)$. We denote its subset of Hilbert–Schmidt (HS) operators as $\mathrm{HS}(\mathscr{X}_1, \mathscr{X}_2)$. We define $B(\mathscr{X}) := B(\mathscr{X}, \mathscr{X})$ and similar for HS operators. We denote the set of positive, self-adjoint, and trace class operators on $\mathscr{X}$ as $B_1^+(\mathscr{X})$. When $\mathscr{X}$ is a finite-dimensional vector space, $B_1^+(\mathscr{X})$ consists of symmetric positive definite matrices.

- We denote the inner-product and norm weighted by a positive and self-adjoint operator $\mathcal{A} : \mathscr{X} \to \mathscr{X}$ as $\langle x_1, x_2\rangle_{\mathcal{A}} := \langle \mathcal{A}^{1/2}x_1, \mathcal{A}^{1/2}x_2\rangle_{\mathscr{X}}$ and $\|x\|_{\mathcal{A}} := \sqrt{\langle \mathcal{A}^{1/2}x, \mathcal{A}^{1/2}x\rangle_{\mathscr{X}}}$ and $\mathcal{A}^{1/2}$ denotes the self-adjoint square root of $\mathcal{A}$. We note that the operator square root is not required in numerical implementation.

- The set of probability distributions defined using Borel σ-algebra on $\mathscr{X}$ is denoted by $\mathscr{P}(\mathscr{X})$. The density between two probability distributions (i.e., Radon–Nikodym derivative) $\mu_1, \mu_2 \in \mathscr{P}(\mathscr{X})$ evaluated at $x \in \mathscr{X}$ is given by $(\mathrm{d}\mu_1/\mathrm{d}\mu_2)(x)$. For probability distributions on finite-dimensional vector spaces, we do not distinguish between a distribution μ and its probability density function $\pi(\boldsymbol{x}) = (\mathrm{d}\mu/\mathrm{d}\mu_L)(\boldsymbol{x})$, where μ_L is the Lebesgue measure. We use $\boldsymbol{x} \sim \pi$ and $\boldsymbol{x} \overset{\text{i.i.d.}}{\sim} \pi$ to denote a π-distributed random variable and independent and identically distributed samples from π, respectively.

- We denote the diffeomorphism group of $\mathscr{X}$ as $\mathrm{Diff}^1(\mathscr{X}) := \big\{\mathcal{T} : \mathscr{X} \to \mathscr{X} \mid \mathcal{T} \text{ is an automorphism, and } \mathcal{T}, \mathcal{T}^{-1} \in C^1(\mathscr{X})\big\}$. For $\mathcal{T} \in \mathrm{Diff}^1(\mathscr{X})$, we denote by $\mathcal{T}_\sharp\mu$ and $\mathcal{T}^\sharp\mu$ the pushforward and pullback of probability distributions in the sense that $\mathcal{T}_\sharp\mu = \mu \circ \mathcal{T}^{-1}$ and $\mathcal{T}^\sharp\mu = \mu \circ \mathcal{T}$.

1.6. Outline of the paper

The remainder of this work proceeds as follows: In Section 2, we introduce the lazy map variational inference method for solving high-dimensional Bayesian inversion. Then, in Section 3, we introduce the optimized surrogate construction for amortized Bayesian inverse for lazy map variational inference. In Section 4, we describe the `LazyDINO` algorithm in detail, including documentation of all offline and online procedures and its role in enabling amortized inference. In Section 5, we define the setup for numerical experiments, including the two infinite-dimensional PDE-constrained BIPs and all metrics utilized to measure

the posterior approximation errors. In Section 6, we present the numerical results and discuss the relative performance of the methods. Finally, we give concluding remarks in Section 7. We have additional results and discussions in the various appendices.

2. Solving Bayesian inverse problems using lazy map variational inference

This section introduces our framework for solving high-dimensional nonlinear BIPs using lazy map variational inference (LMVI). In Section 2.1, we define the setting for BIPs considered in this work. Then, we describe posterior approximation via ridge functions and the resulting inference problems in a latent parameter space of lower dimensions in Sections 2.2 to 2.4. Lastly, we introduce LMVI that approximates the target posterior in the latent space in Sections 2.5 and 2.6.

2.1. Nonlinear Bayesian inverse problems

We denote the unknown parameter of interest $m \in \mathscr{M}$, where $\mathscr{M}$ is a separable Hilbert space with inner product $\langle \cdot, \cdot \rangle_{\mathscr{M}}$. Let $\boldsymbol{y} \in \mathbb{R}^{d_{\boldsymbol{y}}}$ denote observational data, and $\boldsymbol{\mathcal{G}} : \mathscr{M} \to \mathbb{R}^{d_{\boldsymbol{y}}}$ denote a nonlinear parameter-to-observable (PtO) map. We begin with the following standard assumptions.

Assumption 2.1 (Gaussian prior distribution)**.** *We consider a Gaussian prior distribution $\mu = \mathcal{N}(0, \mathcal{C}) \in \mathscr{P}(\mathscr{M})$ with $\mathcal{C} \in B_1^+(\mathscr{M})$.*

Assumption 2.2 (Additive Gaussian noise)**.** *We assume the observed data has the following distribution:*

$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mathcal{G}}(m), \Gamma_n), \tag{4}$$

where $\Gamma_n \in B_1^+(\mathbb{R}^{d_{\boldsymbol{y}}})$.

We consider the inverse problem to recover the parameter m from an observed data vector $\boldsymbol{y}$. We are interested in characterizing the posterior distribution satisfying Bayes' rule, which we denote by $\mu^{\boldsymbol{y}} \in \mathscr{P}(\mathscr{M})$:

$$\frac{\mathrm{d}\mu^{\boldsymbol{y}}}{\mathrm{d}\mu}(m) = \frac{1}{Z^{\boldsymbol{y}}} \exp(-\Phi^{\boldsymbol{y}}(m)), \qquad \Phi^{\boldsymbol{y}}(m) \coloneqq \frac{1}{2} \|\boldsymbol{\mathcal{G}}(m) - \boldsymbol{y}\|^2_{\Gamma_n^{-1}}. \tag{5}$$

Here, we define the *potential* $\Phi^{\boldsymbol{y}} : \mathscr{M} \to \mathbb{R}$ (i.e., the negative log-likelihood), and the normalization constant $Z^{\boldsymbol{y}} \coloneqq \mathbb{E}_{m \sim \mu} [\exp(-\Phi^{\boldsymbol{y}}(m))]$.

Remark 1. *We address two potential concerns regarding our BIP setting. Firstly, to infer parameters with non-Gaussian priors, one can perform inference in transformed coordinates distributed according to a Gaussian prior and subsequently sample from the posterior via the inverse transform; see, e.g., [88, 89]. Secondly, even though we only consider the BIP arising from a single observation, this work straightforwardly extends to a collection of observations, e.g., $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{\mathcal{G}}(m), \Gamma_n)$, in which case the potential in (5) becomes the sum of the negative log-likelihood of each observation. This extension is non-trivial for many AVI methods.*

We define the following Cameron–Martin spaces, separable Hilbert spaces with prior and noise precision–weighted inner products,

$$H_{\mathcal{C}} \coloneqq \{m \in \mathscr{M} : \|m\|_{\mathcal{C}^{-1}} < \infty\}, \quad H_{\Gamma_n} \coloneqq \left\{\boldsymbol{y} \in \mathbb{R}^{d_{\boldsymbol{y}}} : \|\boldsymbol{y}\|_{\Gamma_n^{-1}} < \infty\right\},$$

where $H_{\mathcal{C}}$ and H_{Γ_n} are equipped with inner products $\langle \cdot, \cdot \rangle_{\mathcal{C}^{-1}}$ and $\langle \cdot, \cdot \rangle_{\Gamma_n^{-1}}$, respectively. Note that $H_{\mathcal{C}}$ is continuously embedded in $\mathscr{M}$ and is isomorphic to $\mathscr{M}$ with respect to the identity map only when $\mathscr{M}$ is finite-dimensional.

In this work, we assume the PtO map is H^1_μ-differentiable in the following sense.

Assumption 2.3 (H^1_μ-differentiable PtO map)**.** *We assume the PtO map to live in the Sobolev space with Gaussian measure* $H^1_\mu(\mathscr{M}; H_{\Gamma_n}) \coloneqq \{\boldsymbol{\mathcal{G}} : \mathscr{M} \to \mathbb{R}^{d_{\boldsymbol{y}}} : \|\boldsymbol{\mathcal{G}}\|_{H^1_\mu(\mathscr{M};H_{\Gamma_n})} < \infty\}$ *where*

$$\|\boldsymbol{\mathcal{G}}\|^2_{H^1_\mu(\mathscr{M},H_{\Gamma_n})} \coloneqq \mathbb{E}_{m\sim\mu}\left[\|\boldsymbol{\mathcal{G}}(m)\|^2_{\Gamma_n^{-1}} + \|D_H\boldsymbol{\mathcal{G}}(m)\|^2_{\mathrm{HS}(H_{\mathcal{C}},H_{\Gamma_n})}\right], \tag{6}$$

and $D_H\boldsymbol{\mathcal{G}} : \mathscr{M} \to \mathrm{HS}(H_{\mathcal{C}}, H_{\Gamma_n})$ *is the stochastic derivative or the Malliavin derivative of* $\boldsymbol{\mathcal{G}}$ *that satisfies*

$$\lim_{t\to 0} \left\| t^{-1}\left(\boldsymbol{\mathcal{G}}(m + t\delta m) - \boldsymbol{\mathcal{G}}(m)\right) - D_H\boldsymbol{\mathcal{G}}(m)\delta m\right\|_{\Gamma_n^{-1}} = 0 \quad \forall \delta m \in H_{\mathcal{C}} \quad \mu\text{-a.e.} \tag{7}$$

If the Fréchet derivative $D\boldsymbol{\mathcal{G}} : \mathscr{M} \to \mathrm{HS}(\mathscr{M}, \mathbb{R}^{d_{\boldsymbol{y}}})$ exists, we have the following equality μ-a.e.:

$$D_H\boldsymbol{\mathcal{G}}(m) = D\boldsymbol{\mathcal{G}}(m)|_{H_{\mathcal{C}}}, \quad D_H\boldsymbol{\mathcal{G}}(m)^* = \mathcal{C}D\boldsymbol{\mathcal{G}}(m)^*\Gamma_n^{-1}, \tag{8}$$

where $|_{H_{\mathcal{C}}}$ denotes the restriction of the function domain from $\mathscr{M}$ to $H_{\mathcal{C}}$. We do not distinguish between $D\boldsymbol{\mathcal{G}}(m)$ and $D_H\boldsymbol{\mathcal{G}}(m)$ when it is clear that the domain is $H_{\mathcal{C}}$.

2.2. Subspace decomposition of Bayesian inverse problems

Let $\mathcal{P} \in B(\mathscr{M})$ be a rank-d_r linear projection. We have the corresponding unique decomposition of $\mathscr{M}$ into the image (i.e., the range space) and kernel (i.e., the null space) of $\mathcal{P}$:

$$\mathscr{M} = \mathrm{Im}(\mathcal{P}) \oplus \mathrm{Ker}(\mathcal{P}), \quad m = \underbrace{\mathcal{P}m}_{m_r} + \underbrace{(\mathrm{Id}_{\mathscr{M}} - \mathcal{P})m}_{m_\perp} \quad \forall m \in \mathscr{M},$$

where $\oplus$ denotes the direct sum as defined above. We denote the prior and posterior marginals in $\mathrm{Im}(\mathcal{P})$ by the pushforward $\mu_r \coloneqq \mathcal{P}_\sharp\mu$ and $\mu_r^{\boldsymbol{y}} \coloneqq \mathcal{P}_\sharp\mu^{\boldsymbol{y}}$, respectively. A probability distribution on $\mathscr{M}$ can be decomposed into its marginal probability in $\mathrm{Im}(\mathcal{P})$ and its conditional probability in $\mathrm{Ker}(\mathcal{P})$ in the following sense. For any measurable subset $\mathscr{A} \subseteq \mathscr{M}$ and its decomposition $\mathscr{A} = \mathscr{A}_r \oplus \mathscr{A}_\perp$, where $\mathscr{A}_r \subseteq \mathrm{Im}(\mathcal{P})$ and $\mathscr{A}_\perp \subseteq \mathrm{Ker}(\mathcal{P})$, the prior and posterior probability concentrations on $\mathscr{A}$ are given by

$$\mu(\mathscr{A}) = \int_{\mathscr{A}_r} \mu_{\perp|r}(\mathscr{A}_\perp|m_r)\mathrm{d}\mu_r(m_r), \quad \mu^{\boldsymbol{y}}(\mathscr{A}) = \int_{\mathscr{A}_r} \mu^{\boldsymbol{y}}_{\perp|r}(\mathscr{A}_\perp|m_r)\mathrm{d}\mu^{\boldsymbol{y}}_r(m_r), \tag{9}$$

where $\mu_{\perp|r}(\cdot|m_r), \mu^{\boldsymbol{y}}_{\perp|r}(\cdot|m_r) \in \mathscr{P}(\mathrm{Ker}(\mathcal{P}))$ are the prior and posterior conditionals in $\mathrm{Ker}(\mathcal{P})$. In particular, the prior marginal μ_r, hereafter referred to as the *subspace prior*, and conditional $\mu_{\perp|r}(\cdot|m_r)$ has closed forms given by:

$$\mu_r = \mathcal{N}\left(0, \mathcal{P}\mathcal{C}\mathcal{P}^*\right), \quad \mu_{\perp|r}(\cdot|m_r) = \mathcal{N}\left(\mathcal{C}\mathcal{P}^*(\mathcal{P}\mathcal{C}\mathcal{P}^*)^{-1}m_r - m_r, \mathcal{C}\mathcal{P}^* - \mathcal{P}\mathcal{C}\right), \tag{10}$$

where $\mathcal{P}^*$ is the Hermitian adjoint of $\mathcal{P}$. These forms can be simplified for specific choices of $\mathcal{P}$, which is discussed in Section 2.4.

2.3. Posterior approximation using ridge functions

We proceed under the assumption that the projection $\mathcal{P}$ has been chosen such that the data $\boldsymbol{y}$ are uninformative of the parameter in $\mathrm{Ker}(\mathcal{P})$, i.e., the difference between the prior and the posterior is small in $\mathrm{Ker}(\mathcal{P})$. The process for choosing $\mathcal{P}$ will be delineated in Section 3. Under this assumption, we consider a ridge function approximation of the PtO map:

$$\widetilde{\boldsymbol{\mathcal{G}}} : \mathrm{Im}(\mathcal{P}) \to \mathbb{R}^{d_{\boldsymbol{y}}}, \quad \widetilde{\boldsymbol{\mathcal{G}}} \circ \mathcal{P} \approx \boldsymbol{\mathcal{G}}. \tag{11}$$

An example of such a ridge function is the conditional expectation of the PtO map, where the projected parameter input is lifted into the full space $\mathscr{M}$ by filling $\mathrm{Ker}(\mathcal{P})$ with the prior conditional:

$$\widetilde{\boldsymbol{\mathcal{G}}}_{\mathrm{opt}}(\mathcal{P}m) \coloneqq \mathbb{E}_{m_\perp\sim\mu_{\perp|r}(\cdot|m_r)}\left[\boldsymbol{\mathcal{G}}(\mathcal{P}m + m_\perp)\right]. \tag{12}$$

For a given projection, this ridge function is optimal with respect to the Bochner norm on $L^2_\mu(\mathscr{M}; H_{\Gamma_n})$ [29, 32],

$$\mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{\mathcal{G}}(m)-\widetilde{\boldsymbol{\mathcal{G}}}_{\text{opt}}\left(\mathcal{P}m\right)\right\|^2_{\Gamma_n^{-1}}\right]=\inf_{\widetilde{\boldsymbol{\mathcal{G}}}:\text{Im}(\mathcal{P})\to\mathbb{R}^{d_{\boldsymbol{y}}}}\mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{\mathcal{G}}(m)-\widetilde{\boldsymbol{\mathcal{G}}}(\mathcal{P}m)\right\|^2_{\Gamma_n^{-1}}\right].$$

We refer to $\widetilde{\boldsymbol{\mathcal{G}}}_{\text{opt}}\circ\mathcal{P}$ as the *optimal ridge function*.

A ridge function approximation of the PtO map induces an approximate posterior $\widetilde{\mu}^{\boldsymbol{y}}\in\mathscr{P}(\mathscr{M})$ given by

$$\widetilde{\Phi}^{\boldsymbol{y}}(\mathcal{P}m):=\frac{1}{2}\left\|\widetilde{\boldsymbol{\mathcal{G}}}(\mathcal{P}m)-\boldsymbol{y}\right\|^2_{\Gamma_n^{-1}},\quad \frac{\mathrm{d}\widetilde{\mu}^{\boldsymbol{y}}}{\mathrm{d}\mu}(m)=\frac{1}{\widetilde{Z}^{\boldsymbol{y}}}\exp\left(-\widetilde{\Phi}^{\boldsymbol{y}}(\mathcal{P}m)\right),\tag{13}$$

where $\widetilde{\Phi}^{\boldsymbol{y}}\circ\mathcal{P}\approx\Phi^{\boldsymbol{y}}$. Since the ridge function does not act in $\text{Ker}(\mathcal{P})$, the approximate posterior conditional $\widehat{\mu}^{\boldsymbol{y}}_{\perp|r}$ is proportional to the prior conditional $\mu_{\perp|r}$, and the following holds by Bayes' rule

$$\frac{\mathrm{d}\widetilde{\mu}^{\boldsymbol{y}}_r}{\mathrm{d}\mu_r}(m_r)=\frac{1}{\widetilde{Z}^{\boldsymbol{y}}_r}\exp(-\widetilde{\Phi}^{\boldsymbol{y}}(m_r)),\quad \widetilde{\mu}^{\boldsymbol{y}}_{\perp|r}=\frac{\widetilde{Z}^{\boldsymbol{y}}_r}{\widetilde{Z}^{\boldsymbol{y}}}\mu_{\perp|r},\tag{14}$$

where the *subspace posterior*, $\widetilde{\mu}^{\boldsymbol{y}}_r\in\mathscr{P}(\text{Im}(\mathcal{P}))$, is a marginal of $\widetilde{\mu}^{\boldsymbol{y}}$.

The quality of the ridge function $\widetilde{\boldsymbol{\mathcal{G}}}\circ\mathcal{P}$ can be understood through statistical distances between the posterior μ and the approximate posterior $\widetilde{\mu}^{\boldsymbol{y}}$; these results are covered in Section 3.1.

2.4. Latent Bayesian inverse problems induced by ridge functions

Let $\Psi_r=\{\psi_j\in\mathscr{M}\}_{j=1}^{d_r}$ denote a basis for $\text{Im}(\mathcal{P})$, i.e., $\text{span}(\Psi_r)=\text{Im}(\mathcal{P})$. We refer to Ψ_r as a *reduced basis* on $\mathscr{M}$. The reduced basis defines an encoder $\mathcal{E}_r$ and decoder $\mathcal{D}_r$ pair:

$$\begin{cases}\mathcal{E}_r:\mathscr{M}\ni\sum_{j=1}^{d_r}\boldsymbol{x}_j\psi_j+m_\perp\mapsto\boldsymbol{x}\in\mathbb{R}^{d_r},\\ \mathcal{D}_r:\mathbb{R}^{d_r}\ni\boldsymbol{x}\mapsto\sum_{j=1}^{d_r}\boldsymbol{x}_j\psi_j\in\text{Im}(\mathcal{P}),\end{cases}\qquad\begin{cases}\mathcal{P}=\mathcal{D}_r\circ\mathcal{E}_r,\\ \text{Id}_{d_r}=\mathcal{E}_r\circ\mathcal{D}_r,\end{cases}\tag{15}$$

where Id_{d_r} is the identity matrix in $\mathbb{R}^{d_r}$. We refer to $\mathbb{R}^{d_r}=\mathcal{E}_r(\mathscr{M})$ as the *latent parameter vector space*. The *latent prior* $\pi\in\mathscr{P}(\mathbb{R}^{d_r})$ and *latent posterior* $\widetilde{\pi}^{\boldsymbol{y}}\in\mathscr{P}(\mathbb{R}^{d_r})$ are defined as the pushforward of the prior marginal μ_r and the subspace posterior $\widetilde{\mu}^{\boldsymbol{y}}_r$, respectively, by the encoder $\mathcal{E}_r$, and they satisfy Bayes' rule of the latent parameters:

$$\begin{cases}\pi:=\mathcal{E}_{r\sharp}\mu_r=\mathcal{N}(0,\mathcal{E}_r\mathcal{C}\mathcal{E}_r^*)\\ \widetilde{\pi}^{\boldsymbol{y}}:=\mathcal{E}_{r\sharp}\widetilde{\mu}^{\boldsymbol{y}}_r\end{cases},\quad \widetilde{\pi}^{\boldsymbol{y}}(\boldsymbol{x})=\frac{1}{\widetilde{Z}^{\boldsymbol{y}}_r}\exp\left(-\widetilde{\Phi}^{\boldsymbol{y}}(\mathcal{D}_r\boldsymbol{x})\right)\pi(\boldsymbol{x}).\tag{16}$$

Given latent posterior samples $\boldsymbol{x}^{(j)}\overset{\text{i.i.d.}}{\sim}\widetilde{\pi}^{\boldsymbol{y}}$ and prior conditional samples $m_\perp^{(j)}\overset{\text{i.i.d.}}{\sim}\mu_{\perp|r}(\cdot|\mathcal{D}_r\boldsymbol{x}^{(j)})$, we obtain approximate posterior samples $m^{(j)}\overset{\text{i.i.d.}}{\sim}\widetilde{\mu}^{\boldsymbol{y}}$, where $m^{(j)}=\mathcal{D}_r\boldsymbol{x}^{(j)}+m_\perp^{(j)}$.

While there are many choices of reduced basis, we consider a class of $H_{\mathcal{C}}$–orthonormal reduced basis given as follows:

$$\langle\psi_j,\psi_k\rangle_{\mathcal{C}^{-1}}=\delta_{jk},\quad j,k=1,\dots,d_r,\quad \mathcal{E}_r m=\sum_{j=1}^{d_r}\langle m,\psi_j\rangle_{\mathcal{C}^{-1}}\,\boldsymbol{e}_j,\tag{17}$$

where δ_{jk} is the Kronecker delta, and $\boldsymbol{e}_j$ is the unit vector in the latent space $\mathbb{R}^{d_r}$. Through the definition of the Hermitian adjoint on $\mathscr{M}$, we have $\mathcal{E}_r^*=\mathcal{C}^{-1}\mathcal{D}_r$ and $\mathcal{D}_r^*=\mathcal{E}_r\mathcal{C}$, which implies $\mathcal{E}_r\mathcal{C}\mathcal{E}_r^*=\text{Id}_{\mathbb{R}^{d_r}}$ and $\mathcal{P}^*=\mathcal{C}^{-1}\mathcal{P}\mathcal{C}$ due to (15). Consequently, (10) and (16) yield

$$\pi=\mathcal{N}(\mathbf{0},\text{Id}_{\mathbb{R}^{d_r}}),\qquad\text{(whitened latent prior)}\tag{18}$$

$$\mu_{\perp|r}(\cdot|m_r)\equiv\mu_\perp,\qquad\text{(independence of marginals)}\tag{19}$$

where $\mu_\perp=(\text{Id}_{\mathscr{M}}-\mathcal{P})_\sharp\mu$ is the prior marginal in $\text{Ker}(\mathcal{P})$. As a result of (19), sampling of the conditional $m_\perp^{(j)}\overset{\text{i.i.d.}}{\sim}\mu_{\perp|r}(\cdot|m_r^{(j)})$ can be accomplished via sampling the full prior:

$$m_\perp^{(j)}=m_{\text{pr}}^{(j)}-\mathcal{P}m_{\text{pr}}^{(j)},\quad m_{\text{pr}}^{(j)}\sim\mu.\tag{20}$$

2.5. Lazy map variational inference

We consider TMVI that seeks a diffeomorphic deterministic coupling between a target and a reference distribution [74]. For our BIP in (5), we aim to find a transport map $\mathcal{T} \in \mathrm{Diff}^1(\mathscr{M})$ that couples our reference, the prior $\mu \in \mathscr{P}(\mathscr{M})$, to the target, the posterior $\mu^{\boldsymbol{y}} \in \mathscr{P}(\mathscr{M})$:

$$\mathcal{T}_\sharp \mu = \mu^{\boldsymbol{y}}, \quad \mathcal{T}^\sharp \mu^{\boldsymbol{y}} = \mu.$$

Given $\mathcal{T}$, sampling from the posterior $m_{\text{post}}^{(j)} \overset{\text{i.i.d.}}{\sim} \mu^{\boldsymbol{y}}$ is accomplished by evaluating $m_{\text{post}}^{(j)} = \mathcal{T}(m_{\text{pr}}^{(j)})$, where $m_{\text{pr}}^{(j)} \overset{\text{i.i.d.}}{\sim} \mu$. Since $\mathcal{T}$ is typically unavailable in closed form for nonlinear BIPs, we consider classes of transport maps parametrized by weights $\boldsymbol{\theta} \in \mathbb{R}^{d_\theta}$ such that $\mathcal{T}_{\boldsymbol{\theta}\sharp}\mu$ and μ are mutually absolutely continuous. These weights are found via the solution of a stochastic optimization problem, whose goal is to find $\mathcal{T}_{\boldsymbol{\theta}\sharp}\mu \approx \mu^{\boldsymbol{y}}$. The reverse Kullback-Leibler (rKL) divergence of the approximating distribution from the target posterior $\mu^{\boldsymbol{y}}$

$$\mathcal{D}_{\mathrm{KL}}(\mathcal{T}_{\boldsymbol{\theta}\sharp}\mu || \mu^{\boldsymbol{y}}) = \mathcal{D}_{\mathrm{KL}}(\mu || \mathcal{T}_{\boldsymbol{\theta}}{}^\sharp \mu^{\boldsymbol{y}}) = \mathbb{E}_{m\sim\mu}\left[\log\left(\frac{\mathrm{d}\mu}{\mathrm{d}(\mu^{\boldsymbol{y}} \circ \mathcal{T}_{\boldsymbol{\theta}})}(m)\right)\right]$$

is often used to measure the error of transport map posterior approximation and thereby employed as the objective function for optimization [74, 90]. This objective is equivalent to the *evidence lower bound* objective function.

To make TMVI tractable when $\mathscr{M}$ has high or infinite dimensions, we use *lazy maps*, proposed in [13], which leverages the subspace decomposition as in Section 2.4:

$$\mathbf{T}_{\boldsymbol{\theta}} \coloneqq \mathbf{T}(\cdot, \boldsymbol{\theta}) \in \mathscr{T} \subset \mathrm{Diff}^1(\mathbb{R}^{d_r}), \quad \mathcal{T}_{\boldsymbol{\theta}} \coloneqq \overbrace{(\mathrm{Id}_{\mathscr{M}} - \mathcal{P})}^{\text{identity in } \mathrm{Ker}(\mathcal{P})} + \underbrace{\mathcal{D}_r \circ \mathbf{T}_{\boldsymbol{\theta}} \circ \mathcal{E}_r}_{\text{nonlinear transport in } \mathrm{Im}(\mathcal{P})}, \qquad \text{(Lazy Map)} \tag{21}$$

where a latent space nonlinear transport is used to represent the coupling of the prior and the posterior in $\mathrm{Im}(\mathcal{P})$ while the prior is preserved in $\mathrm{Ker}(\mathcal{P})$. Here we consider a parametrized class $\mathscr{T} \subset \mathrm{Diff}^1(\mathbb{R}^{d_r})$ with weights $\boldsymbol{\theta} \subseteq \mathbb{R}^{d_{\boldsymbol{\theta}}}$ that allows $\mathbf{T}_{\boldsymbol{\theta}\sharp}\pi$ and π to be mutually absolutely continuous and assume the diffeomorphic property is achieved by constraining the map to satisfy $\det \nabla_{\boldsymbol{z}} \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}) > 0$ a.e.; see [74].

The following proposition shows an equivalence in rKL between a lazy map defined on the parameter space $\mathscr{M}$, and a transport map defined on the latent space $\mathbb{R}^{d_r}$ through the optimal ridge function. (16)

Proposition 2.1. *Given a linear projection $\mathcal{P}$ defined using a $H_{\mathcal{C}}$-orthonormal reduced basis and a latent space transport $\mathbf{T}_{\boldsymbol{\theta}} \in \mathscr{T}$, we have*

$$\begin{aligned}\mathcal{D}_{\mathrm{KL}}(\mathcal{T}_{\boldsymbol{\theta}\sharp}\mu || \mu^{\boldsymbol{y}}) &= \mathcal{D}_{\mathrm{KL}}(\mathbf{T}_{\boldsymbol{\theta}\sharp}\pi || \widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}}) + C_1 \\ &= \mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left(\widetilde{\Phi}_{\text{opt}}^{\boldsymbol{y}} \circ \mathcal{D}_r \circ \mathbf{T}_{\boldsymbol{\theta}}\right)(\boldsymbol{z}) + \frac{1}{2}\left\|\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})\right\|^2 - \log\det \nabla_{\boldsymbol{z}} \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})\right] + C_2,\end{aligned} \tag{22}$$

where $\mathcal{T}_{\boldsymbol{\theta}}$ is the lazy map in (21), $\pi = \mathcal{N}(\mathbf{0}, \mathrm{Id}_{\mathbb{R}^{d_r}})$ is the whitened latent prior in (18), $\widetilde{\Phi}_{\text{opt}}^{\boldsymbol{y}}$ and $\widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}}$ are the approximate potential and latent posterior induced by $\widetilde{\mathcal{G}}_{\text{opt}} \circ \mathcal{P}$ in (12), and C_1 and C_2 are constants that do not depend on $\boldsymbol{\theta}$.

The proof of Proposition 2.1 is provided in Appendix C. Due to this result, we may formulate LMVI as the following optimization problem with an equivalent latent representation

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^{\boldsymbol{\theta}}} \mathcal{D}_{\mathrm{KL}}(\mathcal{T}_{\boldsymbol{\theta}\sharp}\mu || \mu^{\boldsymbol{y}}), \qquad \text{(Lazy map variational inference)} \tag{23a}$$

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^{\boldsymbol{\theta}}} \mathcal{D}_{\mathrm{KL}}(\mathbf{T}_{\boldsymbol{\theta}\sharp}\pi || \widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}}). \qquad \text{(Equivalent latent representation)} \tag{23b}$$

The latent representation can be treated as a TMVI problem that seeks $\mathbf{T} \in \mathrm{Diff}^1(\mathbb{R}^{d_r})$ defined as follows.

$$\mathbf{T}_\sharp \pi = \widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}}, \quad \mathbf{T}_\sharp \pi(\boldsymbol{x}) = (\pi \circ \mathbf{T}^{-1})(\boldsymbol{x}) |\det \nabla \mathbf{T}^{-1}(\boldsymbol{x})|, \qquad \text{(Latent space pushforward)}$$

$$\mathbf{T}^\sharp \widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}} = \pi, \quad \mathbf{T}^\sharp \widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}}(\boldsymbol{z}) = (\widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}} \circ \mathbf{T})(\boldsymbol{z}) |\det \nabla \mathbf{T}(\boldsymbol{z})|. \qquad \text{(Latent space pullback)}$$

Given such a transport map, sampling $\boldsymbol{x}^{(j)} \overset{\text{i.i.d.}}{\sim} \widetilde{\pi}^{\boldsymbol{y}}_{\text{opt}}$ is accomplished by evaluating $\boldsymbol{x}^{(j)} = \mathbf{T}(\boldsymbol{z}^{(j)})$, where $\boldsymbol{z}^{(j)} \overset{\text{i.i.d.}}{\sim} \pi$. In turn, approximate posterior, or *pushforward*, samples $m^{(j)} \overset{\text{i.i.d.}}{\sim} \widetilde{\mu}^{\boldsymbol{y}}_{\text{opt}}$ can be drawn simply by lifting $\boldsymbol{x}^{(j)}$ into $\mathscr{M}$ following (20).

Table 1: A summary of BIP problems discussed in Section 2.

BIP Name	Approximation		Parameter space	Prior and Posterior	
Original	None		$\mathscr{M}$	$\mu, \mu^{\boldsymbol{y}}$	(5)
Subspace	$\boldsymbol{\mathcal{G}} \approx \widetilde{\boldsymbol{\mathcal{G}}} \circ \mathcal{P}$	(11)	$\text{Im}(\mathcal{P})$	$\mu_r, \widetilde{\mu}^{\boldsymbol{y}}_r$	(14)
Latent	$\boldsymbol{\mathcal{G}} \approx \widetilde{\boldsymbol{\mathcal{G}}} \circ \mathcal{P}$ $\mathcal{P} = \mathcal{D}_r \circ \mathcal{E}_r$	(11) (15)	$\mathbb{R}^{d_r}$	$\pi, \widetilde{\pi}^{\boldsymbol{y}}$	(16)
Lazy map latent representation	$\boldsymbol{\mathcal{G}} \approx \widetilde{\boldsymbol{\mathcal{G}}}_{\text{opt}} \circ \mathcal{P}$ $\mathcal{P} = \mathcal{D}_r \circ \mathcal{E}_r$	(12) (15)	$\mathbb{R}^{d_r}$	$\pi, \widetilde{\pi}^{\boldsymbol{y}}_{\text{opt}}$	(16)

2.6. Stochastic optimization of lazy map and challenges

For a given transport map parametrization $\mathbf{T}_{\boldsymbol{\theta}} \in \mathscr{T}$, the map parameter vector $\boldsymbol{\theta}$ are typically found via gradient-based stochastic optimization, which in turn requires evaluating Monte Carlo (MC) estimates of the gradient of the objective with respect to $\boldsymbol{\theta}$. The shifted rKL objective, denoted as $\mathcal{L}^{\boldsymbol{y}} : \mathbb{R}^{d_{\boldsymbol{\theta}}} \to \mathbb{R}$, can be expressed as an expectation of a single-sample MC estimator, $\mathcal{L}^{\boldsymbol{y}}_1 : \mathscr{M} \times \mathbb{R}^{d_{\boldsymbol{\theta}}} \to \mathbb{R}$, defined as follows:

$$\mathcal{L}^{\boldsymbol{y}}(\theta) := \mathbb{E}_{m\sim\mu}\left[\mathcal{L}^{\boldsymbol{y}}_1(m, \boldsymbol{\theta})\right] := \mathcal{D}_{\text{KL}}(\mathcal{T}_{\boldsymbol{\theta}\sharp}\mu || \mu^{\boldsymbol{y}}) - C_2, \tag{24}$$

$$\mathcal{L}^{\boldsymbol{y}}_1(\mathcal{D}_r\boldsymbol{z}^{(j)} + m^{(j)}_\perp, \theta) = \Phi^{\boldsymbol{y}}\left((\mathcal{D}_r \circ \mathbf{T}_\theta)(\boldsymbol{z}) + m^{(j)}_\perp\right) + \frac{1}{2}\left\|\mathbf{T}_\theta(\boldsymbol{z}^{(j)})\right\|^2 - \log\det\nabla_{\boldsymbol{z}}\mathbf{T}_\theta(\boldsymbol{z}^{(j)}), \tag{25}$$

where $\boldsymbol{z}^{(j)} \overset{\text{i.i.d.}}{\sim} \pi$ and $m^{(j)}_\perp \overset{\text{i.i.d.}}{\sim} \mu_\perp$. The single-sample MC gradient estimator with respect to the map parameters $\boldsymbol{\theta}$ takes the following form

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{y}}_1(\mathcal{D}_r\boldsymbol{z}^{(j)} + m^{(j)}_\perp, \theta) = \nabla_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}^{(j)})^\top(\mathcal{E}_r \circ D_H\Phi^{\boldsymbol{y}})\left((\mathcal{D}_r \circ \mathbf{T})(\boldsymbol{z}^{(j)}) + m^{(j)}_\perp\right)\\ + \nabla_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}^{(j)})^\top\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}^{(j)}) - \nabla_{\boldsymbol{\theta}}(\log\det\nabla_{\boldsymbol{z}}\mathbf{T}_\theta)(\boldsymbol{z}^{(j)}),\end{aligned} \tag{26}$$

where $D_H\Phi^{\boldsymbol{y}}(m) \coloneqq D_H\boldsymbol{\mathcal{G}}(m)^*(\boldsymbol{\mathcal{G}}(m) - \boldsymbol{y})$ is the prior-preconditioned gradient of the potential; see (8).

The MC gradient estimator of the rKL objective is then

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}) \approx \nabla_{\boldsymbol{\theta}}\widehat{\mathcal{L}}^{\boldsymbol{y}}(\boldsymbol{\theta}) = \frac{1}{N_{\text{MC}}}\sum_{j=1}^{N_{\text{MC}}}\nabla_{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{y}}_1(m^{(j)}, \boldsymbol{\theta}), \quad m^{(j)} \overset{\text{i.i.d.}}{\sim} \mu.$$

Remark 2. *In the numerical results, starting in Section 6, the lazy map optimization is implemented with an alternative form of the rKL objective where samples from $\mu_\perp$ are estimated as $\mathbb{E}[\mu_\perp] = 0$:*

$$\mathcal{L}^{\boldsymbol{y}}(\theta) := \mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\mathcal{L}^{\boldsymbol{y}}_1(\mathcal{D}_r\boldsymbol{z}, \boldsymbol{\theta})\right].$$

This leads to a different TMVI problem induced by the ridge function $\boldsymbol{\mathcal{G}} \circ \mathcal{P}$ instead of $\widetilde{\boldsymbol{\mathcal{G}}}_{\text{opt}} \circ \mathcal{P}$ in Proposition 2.1. We empirically found that it performs better under a limited computational budget, likely due to a superior bias–variance trade-off in TMVI.

The computation cost of evaluating the second and third terms in (26) only depends on the parametrization of the transport map. Notably triangular transport maps [74] and parameterization built as compositions of triangular maps (e.g., inverse autoregressive flows [91]), are structured such these terms are efficiently computable. The first term requires evaluating the PtO map and its prior-preconditioned gradient. When the PtO map is expensive to evaluate, as is the case with large-scale PDE-governed problems, optimizing for an accurate lazy map is prohibitively expensive.

3. Optimized surrogate construction for lazy map variational inference

This section discusses the construction of a fast-to-evaluate neural network ridge function surrogate $\widetilde{\boldsymbol{\mathcal{G}}} \circ \mathcal{P} \approx \boldsymbol{\mathcal{G}}$ that leads to a small and controllable expected error in surrogate-driven LMVI, where the expectation is taken over the marginal distribution of data vectors with density $\gamma \in \mathscr{P}(\mathbb{R}^{d_{\boldsymbol{y}}})$ where $\gamma(\boldsymbol{y}) \propto Z^{\boldsymbol{y}}$ as in (4). Our strategy for constructing this surrogate is given as follows.

1. Minimizing an upper bound on $\mathbb{E}_{\boldsymbol{y}\sim\gamma}[\mathcal{D}_{\mathrm{KL}}(\mu^{\boldsymbol{y}}||\widetilde{\mu}^{\boldsymbol{y}})]$, the expected forward KL divergence (fKL) from the posterior $\mu^{\boldsymbol{y}}$ to the approximate posterior defined by the surrogate, $\widetilde{\mu}^{\boldsymbol{y}}$.
2. Minimizing an upper bound on the expected optimality gap $\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\sqrt{\mathcal{L}^{\boldsymbol{y}}(\widetilde{\boldsymbol{\theta}}^{\boldsymbol{y},\dagger}) - \mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}^{\boldsymbol{y},\dagger})}\right]$ for surrogate-driven LMVI, where $\boldsymbol{\theta}^{\boldsymbol{y},\dagger}$ is the true minimizer of the rKL objective and $\widetilde{\boldsymbol{\theta}}^{\boldsymbol{y},\dagger}$ is the minimizer found via the ridge function surrogate.

In this section, we show the resulting ridge function surrogate is `DIPNet` [24] trained using the derivative-informed learning method [30] and it leads to a latent space surrogate rKL objective.

3.1. Error analysis for surrogate-driven lazy map variational inference

Recall the definition of $\widetilde{\boldsymbol{\mathcal{G}}}_{\mathrm{opt}}$ in (12). We define the following finite-dimensional latent representations as follows:

$$\mathbf{g} \coloneqq \boldsymbol{V}^* \circ \widetilde{\boldsymbol{\mathcal{G}}} \circ \mathcal{D}_r : \quad \mathbb{R}^{d_r} \to \mathbb{R}^{d_{\boldsymbol{y}}}, \quad \widetilde{\boldsymbol{\mathcal{G}}} = \boldsymbol{V} \circ \mathbf{g} \circ \mathcal{E}_r : \quad \mathrm{Im}(\mathcal{P}) \to H_{\Gamma_n}, \tag{27a}$$

$$\mathbf{g}_{\mathrm{opt}} \coloneqq \boldsymbol{V}^* \circ \widetilde{\boldsymbol{\mathcal{G}}}_{\mathrm{opt}} \circ \mathcal{D}_r : \mathbb{R}^{d_r} \to \mathbb{R}^{d_{\boldsymbol{y}}}, \quad \widetilde{\boldsymbol{\mathcal{G}}}_{\mathrm{opt}} = \boldsymbol{V} \circ \mathbf{g}_{\mathrm{opt}} \circ \mathcal{E}_r : \mathrm{Im}(\mathcal{P}) \to H_{\Gamma_n}. \tag{27b}$$

Here $\boldsymbol{V} \in \mathrm{HS}(\mathbb{R}^{d_{\boldsymbol{y}}}, H_{\Gamma_n})$ is a full-rank matrix with columns consists of H_{Γ_n}-orthonormal basis and $\boldsymbol{V}^* = \boldsymbol{V}^\top \Gamma_n^{-1}$ is its Hermitian adjoint that satisfies $\boldsymbol{V}^*\boldsymbol{V} = \mathrm{Id}_{d_{\boldsymbol{y}}}$. Note that $\boldsymbol{V}^*$ is a whitening transformation on the data space.

The following theorem provides an upper bound on the expected fKL between the true posterior and ridge function approximated posterior taken over the marginal distribution of data.

Theorem 3.1 (Posterior approximation through a ridge function surrogate)**.** *Given $\boldsymbol{\mathcal{G}} \in H^1_\mu(\mathscr{M}; H_{\Gamma_n})$ and a projector $\mathcal{P} \in B(\mathscr{M})$ defined via an $H_{\mathcal{C}}$-orthonormal reduced basis as in (17), we have the following inequality for the approximate posterior $\widetilde{\mu}^{\boldsymbol{y}}$ in (14) defined via any ridge function $\widetilde{\boldsymbol{\mathcal{G}}} \circ \mathcal{P} \in L^2_\mu(\mathscr{M}; H_{\Gamma_n})$:*

$$\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\mathcal{D}_{\mathrm{KL}}(\mu^{\boldsymbol{y}}||\widetilde{\mu}^{\boldsymbol{y}})\right] \le \underbrace{\mathrm{Tr}_{H_{\mathcal{C}}}\left(\left(\mathrm{Id}_{H_{\mathcal{C}}} - \mathcal{P}\right)\mathcal{H}_A\left(\mathrm{Id}_{H_{\mathcal{C}}} - \mathcal{P}\right)\right)}_{\text{Parameter reduction error}} + \underbrace{\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\mathbf{g}_{\mathrm{opt}}(\boldsymbol{z}) - \mathbf{g}(\boldsymbol{z})\right\|^2\right]}_{\text{Latent representation error}},$$

where $\mathrm{Tr}_{H_{\mathcal{C}}} : B_1^+(H_{\mathcal{C}}) \to \mathbb{R}$ returns the trace of operators on $H_{\mathcal{C}}$, and $\mathcal{H}_A \in B_1^+(H_{\mathcal{C}})$ is the expected prior–preconditioned Gauss–Newton Hessian of the potential:

$$\mathcal{H}_A \coloneqq \mathbb{E}_{m\sim\mu}\left[D_H\boldsymbol{\mathcal{G}}(m)^* D_H\boldsymbol{\mathcal{G}}(m)\right]. \tag{28}$$

Here $D_H\boldsymbol{\mathcal{G}}(m)^* \in \mathrm{HS}(H_{\Gamma_n}, H_{\mathcal{C}})$ denotes the Hermitian adjoint of the stochastic derivative $D_H\boldsymbol{\mathcal{G}}(m)$ in (7).

The bound decomposes the expected error into terms involving the parameter reduction error that depends on the choice of $\mathcal{P}$ and the discrepancy between the surrogate latent representation $\mathbf{g}$ and the optimal latent representation $\mathbf{g}_{\text{opt}}$ of $\widetilde{\mathcal{G}}_{\text{opt}} \circ \mathcal{P}$. The proof of Theorem 3.1 can be found in Appendix D.1 and follows from results in [29, 31, 32].

For surrogate-driven LMVI, understanding the expected discrepancy between the posteriors and its transport targets (i.e., the surrogate approximated posteriors $\widetilde{\mu}^{\boldsymbol{y}}$) is insufficient, as the transport map is constructed through the process of gradient-based stochastic optimization and the accuracy of the rKL gradient approximation is also important. The following theorem establishes error upper bounds for the surrogate objective gradient.

Theorem 3.2 (Surrogate approximation of the rKL objective gradient)**.** *Given $\boldsymbol{\mathcal{G}} \in H^1_\mu(\mathscr{M}; H_{\Gamma_n})$, a linear projector $\mathcal{P} \in B(\mathscr{M})$ defined using an $H_{\mathcal{C}}$-orthonormal reduced basis as in (17). Assume we have a latent space transport $\mathsf{T}_{\boldsymbol{\theta}} \in \mathscr{T}$ with an essentially bounded density between $\mathsf{T}_{\boldsymbol{\theta}\sharp}\pi$ and π and an essentially-bounded Jacobian with respect to $\boldsymbol{\theta}$. We have the following error upper bound for the approximate gradient of rKL objective $\widetilde{\mathcal{L}}^{\boldsymbol{y}}(\boldsymbol{\theta})$ given by a ridge function,*

$$\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\|\nabla_{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}\widetilde{\mathcal{L}}^{\boldsymbol{y}}(\boldsymbol{\theta})\|\right] \lesssim \left(\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\|\mathbf{g}_{\text{opt}}(\boldsymbol{z}) - \mathbf{g}(\boldsymbol{z})\|^2 + \|\nabla\mathbf{g}_{\text{opt}}(\boldsymbol{z}) - \nabla\mathbf{g}(\boldsymbol{z})\|_F^2\right]\right)^{1/2}, \tag{29}$$

where $\lesssim$ denotes bounded up to a multiplicative constant.

The proof of Theorem 3.2 is presented in Appendix D.2. Our result states that the expected gradient error is controlled by the latent representation error measured in a π-weighted Sobolev norm on $H^1_\pi(\mathbb{R}^{d_r}; \mathbb{R}^{d_y})$, which additionally contain the expected error in the Jacobian compared to the error measure using π-weighted Bochner norm on $L^2_\pi(\mathbb{R}^{d_r}; \mathbb{R}^{d_y})$ in Theorem 3.1. Notably, the two error measures are generally not equivalent, and the Sobolev norm is stronger than the Bochner norm. This result reflects the fact that the gradient of the rKL objective involves the Jacobian of the PtO map, and the surrogate Jacobian accuracy affects the optimization of lazy maps. To further explore the consequences of surrogate Jacobian misfit, we consider the following corollary on the expected optimality gap for surrogate-driven LMVI under a stronger set of assumptions.

Corollary 3.3 (Optimality gap for surrogate-driven LMVI)**.** *Suppose the assumptions in Theorem 3.2 holds. Let $\boldsymbol{\theta}^{\boldsymbol{y},\dagger}$ and $\widetilde{\boldsymbol{\theta}}^{\boldsymbol{y},\dagger}$ denote γ-measurable functions that return the second order stationary points of $\mathcal{L}^{\boldsymbol{y}}$ and $\widetilde{\mathcal{L}}^{\boldsymbol{y}}$ γ-a.e., respectively. Let $B_r(\boldsymbol{x})$ denote a ball of radius r centered at $\boldsymbol{x}$. We assume that $R^{\boldsymbol{y}}$ and $\lambda^{\boldsymbol{y}}$ are γ-essentially bounded from below by some positive constants such that (i) $\widetilde{\boldsymbol{\theta}}^{\boldsymbol{y},\dagger} \in B_{R^{\boldsymbol{y}}}(\boldsymbol{\theta}^{\boldsymbol{y},\dagger})$ γ-a.e., and (ii) $\nabla^2_{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}) \succeq \lambda^{\boldsymbol{y}}\text{Id}_{\mathbb{R}^{d_r}}$ for all $\boldsymbol{\theta} \in B_{R^{\boldsymbol{y}}}(\boldsymbol{\theta}^{\boldsymbol{y},\dagger})$ γ-a.e.*

We have the following upper bound on the optimality gap:

$$\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\sqrt{\mathcal{L}^{\boldsymbol{y}}(\widetilde{\boldsymbol{\theta}}^{\boldsymbol{y},\dagger}) - \mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}^{\boldsymbol{y},\dagger})}\right] \lesssim \left(\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\|\mathbf{g}_{\text{opt}}(\boldsymbol{z}) - \mathbf{g}(\boldsymbol{z})\|^2 + \|\nabla\mathbf{g}_{\text{opt}}(\boldsymbol{z}) - \nabla\mathbf{g}(\boldsymbol{z})\|_F^2\right]\right)^{1/2}. \tag{30}$$

This result states that if the rKL objective is locally strongly convex near the true rKL minimizers and the minimizers found by the surrogate lands within those locally convex regions, we can bound the expected optimality gap by the latent representation error measured by the weighted Sobolev norm. The assumptions in Corollary 3.3 are commonly employed in the optimization and machine learning literature, either directly via local strong convexity or through the Polyak-Łojasiewicz inequality, see for example [92]. These two results together demonstrate the need to control the surrogate Jacobian error in the context of LMVI.

Motivated by these results, we delineate the procedure for constructing a surrogate model for LMVI in the following subsections.

3.2. Minimize the parameter reduction error: derivative-informed subspace

We seek an $H_{\mathcal{C}}$-orthonormal reduced basis $\{\psi_j\}_{j=1}^{d_r}$ such that $\text{span}(\{\psi_j\}_{j=1}^{d_r}) = \text{Im}(\mathcal{P})$ and the parameter reduction error term in Theorem 3.1 is minimized. This can be accomplished by finding the parameter subspace that the PtO map is most sensitive to in expectation. This subspace is often referred to as

the derivative-informed subspace or active subspace [32], and it can be computed from the dominant d_r eigenbases arising from the following eigenvalue problem in $H_{\mathcal{C}}$,

$$\mathcal{H}_A \psi_j = \lambda_j \psi_j, \quad \langle \psi_j, \psi_k \rangle_{\mathcal{C}^{-1}} = \delta_{jk}, \quad \lambda_1 \geq \lambda_2 \geq \cdots \geq 0 \tag{31}$$

where $\mathcal{H}_A$ is the prior-preconditioned Gauss–Newton Hessian in (28). Under a stronger Fréchet differentiability assumption, the eigenvalue problem in $H_{\mathcal{C}}$ is equivalent to a more common form of a generalized eigenvalue problem in $\mathscr{M}$ due to (8):

$$\mathbb{E}_{m\sim\mu}\left[D\boldsymbol{\mathcal{G}}(m)^*\Gamma_n^{-1}D\boldsymbol{\mathcal{G}}(m)\right]\psi_j = \lambda_j \mathcal{C}^{-1}\psi_j, \quad \langle \psi_k, \psi_j \rangle_{\mathcal{C}^{-1}} = \delta_{jk}, \quad \lambda_1 \geq \lambda_2 \geq \cdots \geq 0. \tag{32}$$

The eigenvalue problem in (32) can be found in [31, 32, 36, 37, 93]. The minimum value of the parameter reduction error is

$$\min_{\substack{\mathcal{P}\in\{\text{rank}-d_r \text{ linear} \\ \text{projection on } \mathscr{M}\}}} \text{Tr}_{H_{\mathcal{C}}}\left((\text{Id}_{H_{\mathcal{C}}} - \mathcal{P})\,\mathcal{H}_A\,(\text{Id}_{H_{\mathcal{C}}} - \mathcal{P})\right) = \sum_{j>d_r} \lambda_j. \tag{33}$$

This derivative-based reduced basis leads to an expected parameter reduction error proportional to the eigenvalue tail sum in (31) corresponding to the discarded eigenbases. Existing bounds for the truncated Karhunen–Loéve expansion of the parameter are strictly higher than (33); see [29, 32].

3.3. Minimize the latent representation error: Conventional operator learning

We first consider a neural operator ridge function using a neural network latent representation $\mathbf{g}_{\text{NN}} : \mathbb{R}^{d_r} \times \mathbb{R}^{d_{\boldsymbol{w}}} \to \mathbb{R}^{d_{\boldsymbol{y}}}$:

$$\mathbf{g}_{\boldsymbol{w}}(\boldsymbol{x}) \coloneqq \mathbf{g}_{\text{NN}}(\boldsymbol{x}, \boldsymbol{w}), \qquad \text{(Neural latent representation)} \tag{34a}$$

$$\widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}(\mathcal{P}m) \coloneqq \boldsymbol{V}\mathbf{g}_{\text{NN}}(\mathcal{E}_r m, \boldsymbol{w}), \qquad \text{(Neural operator ridge function)} \tag{34b}$$

where $\boldsymbol{w} \in \mathbb{R}^{d_{\boldsymbol{w}}}$ consists of trainable neural network weights. Neural network surrogates architecture using the derivative-informed subspace (32) are known as `DIPNet` [24].

Motivated by Theorem 3.1, it would be sensible to find the neural network weights by minimizing the latent representation error, which also minimizes the upper bound on the expected surrogate posterior approximation error:

$$\min_{\boldsymbol{w}\in\mathbb{R}^{d_{\boldsymbol{w}}}} \mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\mathbf{g}_{\text{opt}}(\boldsymbol{z}) - \mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|^2\right]. \tag{35}$$

However, estimating this objective function requires a nested MC method due to the $\text{Ker}(\mathcal{P})$ marginalization in $\mathbf{g}_{\text{opt}}$; see definitions in (11) and (27a). Specifically, $\boldsymbol{z}^{(j)} \sim \pi$, $1 \leq j \leq N_{\text{out}}$, are used to estimate the objective, and $m_\perp^{(j,k)} \sim \mu_\perp$, $1 \leq k \leq N_{\text{in}}$, are used to estimate the output of the optimal latent representation at each $\boldsymbol{z}^{(j)}$. The nested MC sample generation requires $N_{\text{out}} \times N_{\text{in}}$ PtO map evaluations. However, the inner MC is unnecessary when $\mathcal{P}$ is chosen as in Section 3.2, since the PtO map is insensitive to changes in $\text{Ker}(\mathcal{P})$; see, e.g., [36, Corollary 7.5]. Therefore, we consider the conventional operator learning method with error measure using the norm on the Bochner space $L^2_\mu(\mathscr{M}, H_{\Gamma_n})$, i.e., a prior-weighted mean squared error objective:

$$\min_{\boldsymbol{w}\in\mathbb{R}^{d_{\boldsymbol{w}}}} \mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{\mathcal{G}}(m) - \widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}(\mathcal{P}m)\right\|^2_{\Gamma_n^{-1}}\right], \qquad \text{(Conventional } L^2_\mu \text{ operator learning)} \tag{36}$$

$$\min_{\boldsymbol{w}\in\mathbb{R}^{d_{\boldsymbol{w}}}} \mathbb{E}_{(\boldsymbol{z},m_\perp)\sim\pi\otimes\mu_\perp}\left[\left\|\underbrace{\boldsymbol{V}^*\boldsymbol{\mathcal{G}}(\mathcal{D}_r\boldsymbol{z} + m_\perp)}_{\approx\,\mathbf{g}_{\text{opt}}(\boldsymbol{z})} - \mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|^2\right]. \qquad \text{(Equivalent latent representation)} \tag{37}$$

The equivalent latent representation of the operator learning objective reveals that this objective can be derived from the neural latent representation error (35) using a single sample ($m_\perp \sim \mu_\perp$) estimate of the

marginalization in $\mathbf{g}_{\text{opt}}$. For notational convenience, we will use $\boldsymbol{g}^{(j)}$ to denote the i.i.d. *whitened PtO samples* used to estimate the conventional L^2_μ operator learning objective:

$$\boldsymbol{g}^{(j)} \coloneqq \boldsymbol{V}^*\boldsymbol{\mathcal{G}}(m^{(j)}) \in \mathbb{R}^{d_{\boldsymbol{y}}}, \quad m^{(j)} \overset{\text{i.i.d.}}{\sim} \mu. \qquad \text{(whitened PtO sample)} \tag{38}$$

We refer to `DIPNet` surrogates trained using the conventional L^2_μ operator learning method in (36) as `RB-NO` (reduced basis neural operator) in contrast to surrogates construction introduced in the following subsection.

3.4. Minimizing the expected optimality gap: Derivative-informed operator learning

While the conventional L^2_μ learning problem presented in Section 3.3 is suitable for constructing a neural operator ridge function, Theorem 3.2 shows that controlling latent representation error in H^1_π controls both the expected gradient error, as well as the expected optimality gap between the exact and surrogate variational inference objective functions. To this end, we consider minimizing the latent representation error measured by the π-weighted Sobolev norm on $H^1_\pi(\mathbb{R}^{d_r}; \mathbb{R}^{d_{\boldsymbol{y}}})$:

$$\min_{\boldsymbol{w}\in\mathbb{R}^{d_{\boldsymbol{w}}}} \mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\mathbf{g}_{\text{opt}}(\boldsymbol{z}) - \mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|^2 + \left\|\nabla\mathbf{g}_{\text{opt}}(\boldsymbol{z}) - \nabla_{\boldsymbol{z}}\mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|^2_F\right]. \tag{39}$$

However, as discussed in the previous subsection, estimating the objective function above also requires a nested MC method. To circumvent this issue, we adopt an derivative-informed H^1_μ operator learning objective following [29, 30]:

$$\begin{aligned}\min_{\boldsymbol{w}\in\mathbb{R}^{d_{\boldsymbol{w}}}} \mathbb{E}_{m\sim\mu}\Big[&\left\|\boldsymbol{\mathcal{G}}(m) - \widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}(\mathcal{P}m)\right\|^2_{\Gamma_n^{-1}} \\ &+ \left\|D\boldsymbol{\mathcal{G}}(m) - D(\widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}\circ\mathcal{P})(m)\right\|^2_{\text{HS}(H_{\mathcal{C}},H_{\Gamma_n})}\Big]\end{aligned} \qquad \text{(Derivative-informed operator learning)} \tag{40}$$

$$\begin{aligned}\min_{\boldsymbol{w}\in\mathbb{R}^{d_{\boldsymbol{w}}}} \mathbb{E}_{(\boldsymbol{z},m_\perp)\sim\pi\otimes\mu_\perp}\Big[&\left\|\boldsymbol{V}^*\boldsymbol{\mathcal{G}}(\mathcal{D}_r\boldsymbol{z}+m_\perp) - \mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|^2 \\ &+ \Big\|\underbrace{\boldsymbol{V}^*\circ D\boldsymbol{\mathcal{G}}(\mathcal{D}_r\boldsymbol{z}+m_\perp)\circ\mathcal{D}_r}_{\approx\nabla\mathbf{g}_{\text{opt}}(\boldsymbol{z})} - \nabla_{\boldsymbol{z}}\mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\Big\|^2_F\Big]\end{aligned} \qquad \text{(Equivalent latent representation)} \tag{41}$$

The equivalent latent representation reveals that the derivative-informed learning objective can be derived from (39) using a single-sample ($m_\perp \sim \mu_\perp$) estimate for the marginalization in both $\mathbf{g}_{\text{opt}}$ and $\nabla_{\boldsymbol{z}}\mathbf{g}_{\text{opt}}$; see Appendix B for a discussion on the marginalization in $\nabla_{\boldsymbol{z}}\mathbf{g}_{\text{opt}}$. We refer to `DIPNet` surrogates trained using the derivative-informed H^1_μ operator learning method as `RB-DINO` (reduced basis derivative-informed neural operator).

We emphasize that one only needs samples of the latent representation of the derivative for H^1_μ operator learning compared to L^2_μ operator learning,

$$\boldsymbol{J}_r^{(j)} \coloneqq \boldsymbol{V}^*\circ D\boldsymbol{\mathcal{G}}(m^{(j)})\circ\mathcal{D}_r \in \mathbb{R}^{d_{\boldsymbol{y}}\times d_r} \quad m^{(j)}\sim\mu. \qquad \text{(whitened latent Jacobian)} \tag{42}$$

For notational convenience, we use $\boldsymbol{J}_r^{(j)}$ to denote the i.i.d. samples of the *whitened latent Jacobian sample* of the PtO map.

3.5. Surrogate-driven lazy map variational inference in the latent space

We use a trained ridge function surrogate $\widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}\circ\mathcal{P}$ to replace the PtO map $\boldsymbol{\mathcal{G}}$ and its Jacobian $D\boldsymbol{\mathcal{G}}$ evaluations during stochastic optimization of the latent space transport $\mathsf{T}_{\boldsymbol{\theta}}$. Specifically, a single-sample

estimate of the surrogate rKL $\widetilde{\mathcal{L}}_1^{\boldsymbol{y}} : \mathscr{M} \times \mathbb{R}^{d_{\boldsymbol{\theta}}} \to \mathbb{R}$, replacing (25), and its gradient, replacing (26), can be equivalently represented in the latent space as $\widetilde{\mathcal{L}}_{1,r}^{\boldsymbol{y}} : \mathbb{R}^{d_r} \times \mathbb{R}^{d_{\boldsymbol{\theta}}} \to \mathbb{R}$:

$$\widetilde{\mathcal{L}}_1^{\boldsymbol{y}}(m, \boldsymbol{\theta}; \boldsymbol{w}) \equiv \widetilde{\mathcal{L}}_{1,r}^{\boldsymbol{y}}(\mathcal{E}_r m, \boldsymbol{\theta}; \boldsymbol{w}) \tag{43a}$$

$$\widetilde{\mathcal{L}}_{1,r}^{\boldsymbol{y}}(\boldsymbol{z}, \boldsymbol{\theta}; \boldsymbol{w}) = \frac{1}{2} \left\| (\mathbf{g}_{\boldsymbol{w}} \circ \mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - \boldsymbol{V}^* \boldsymbol{y} \right\|^2 + \frac{1}{2} \left\| \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}) \right\|^2 - \log\det \nabla_{\boldsymbol{z}} \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}^{(j)}), \tag{43b}$$

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} \widetilde{\mathcal{L}}_{1,r}^{\boldsymbol{y}}(\boldsymbol{z}, \boldsymbol{\theta}; \boldsymbol{w}) = \nabla_{\boldsymbol{\theta}} \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})^\top \left(\nabla_{\boldsymbol{z}} \mathbf{g}_{\boldsymbol{w}} \circ \mathbf{T}_{\boldsymbol{\theta}} \right)(\boldsymbol{z})^\top \left((\mathbf{g}_{\boldsymbol{w}} \circ \mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - \boldsymbol{V}^* \boldsymbol{y} \right) \\ + \nabla_{\boldsymbol{\theta}} \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}^{(j)})^\top \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}^{(j)}) - \nabla_{\boldsymbol{\theta}} (\log\det \nabla_{\boldsymbol{z}} \mathbf{T}_{\theta})(\boldsymbol{z}^{(j)}) \end{aligned} \tag{43c}$$

Consequently, the surrogate-driven training of lazy maps proceeds entirely in the parameter latent space. After the latent space transport map $\mathbf{T}_{\boldsymbol{\theta}}$ is optimized, we use the map to produce latent space posterior samples $\boldsymbol{x}^{(j)} \sim \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z}^{(j)})$, and they can be lifted to the full parameter space via sampling the prior as in (20).

4. The `LazyDINO` method

In this section, we present a high-level overview of the steps involved in `LazyDINO` using schematics and brief descriptions. We refer the reader to Appendix E for a more detailed exposition of these steps.

4.1. Offline phase: `RB-DINO` surrogate construction

In Figure 1, we provide a schematic for the offline surrogate construction. In this phase, one first defines the prior μ and PtO map $\boldsymbol{\mathcal{G}}$ that determines the class of BIPs to be solved by `LazyDINO`. Subsequently, encoders and decoders for the parameter are constructed as delineated in Section 3.2. The training samples are then generated and reduced to their latent representations as in (38) and (42). In particular, the full PtO map Jacobian samples are never formed. Instead, they are compressed matrix-free using the parameter decoder $\mathcal{D}_r$ for more information. The training sample generation is often computationally costly, as the PtO map evaluations often require model solutions, and the PtO map Jacobian actions require computing the forward or adjoint model sensitivity. Once the training samples are collected, a given neural latent representation $\mathbf{g}_{\boldsymbol{w}}$ is trained using the derivative-informed learning method in the latent space (39). We refer to [29, 30] for more implementation details and theory on `RB-DINO` surrogate construction.

4.2. Online phase: Rapid `LazyDINO` transport map construction

Once a `RB-DINO` surrogate PtO map is constructed, it is used in place of the PtO map in the lazy map training, which removes the computational bottleneck of model solutions and makes rapid online inference possible. The process for lazy map construction for a single instance of observational data $\boldsymbol{y}$, is shown in Figure 2. This process involves defining the transport map architecture $\mathbf{T}_{\boldsymbol{\theta}}$, and the associated latent space rKL objective (43a). The transport map is then optimized with respect to its map parameters $\boldsymbol{\theta}$. The trained latent space transport map pushes the whitened latent prior π in (18) to an approximation of the latent posterior induced by the optimal ridge function $\mathbf{T}_{\boldsymbol{\theta}\sharp}\pi \approx \widetilde{\pi}_{\text{opt}}^{\boldsymbol{y}}$ as in Proposition 2.1. These samples can then be decoded to generate samples in the full space using the prior μ.

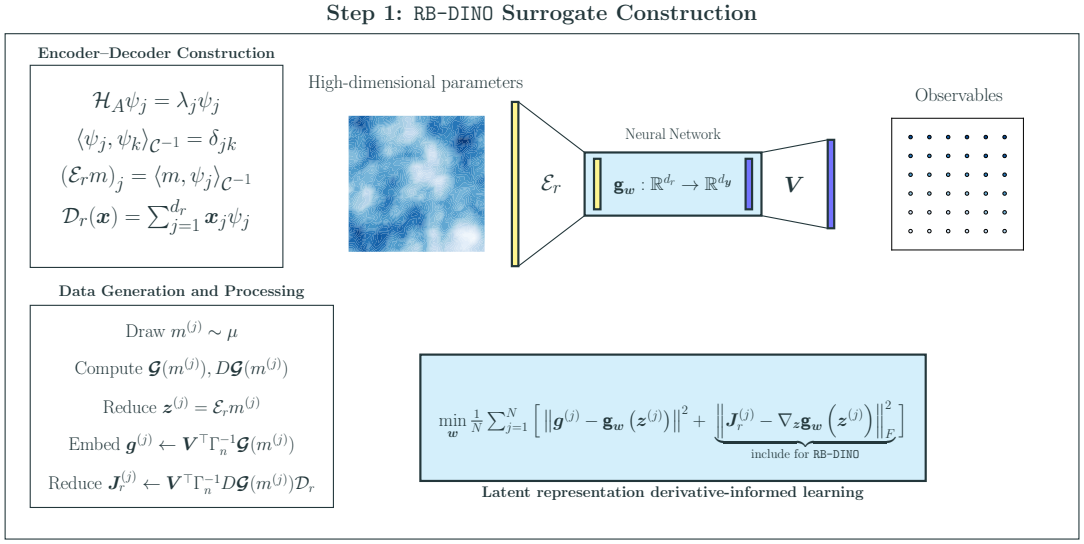

Figure 1: Overview of the RB-DINO construction.

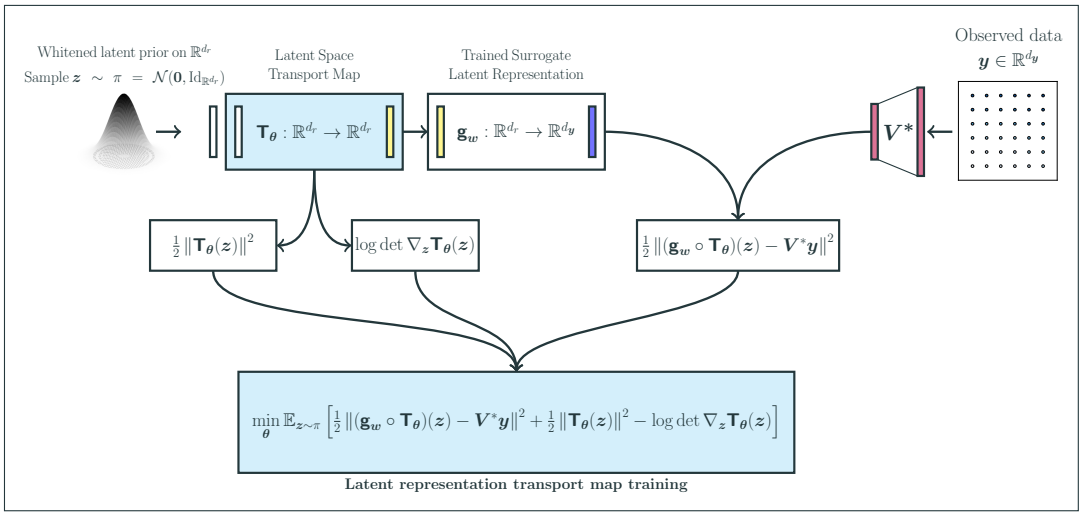

Figure 2: Overview of the latent representation lazy map construction.

A major point of emphasis for LazyDINO is that the computationally expensive aspects of the method are limited to the offline phase. The RB-DINO surrogate replaces the expensive-to-evaluate and often implicitly

defined PtO map with a fast-to-evaluate explicit function. In practice, this leads to potentially enormous speedups for all computations associated with the likelihood and its gradient evaluations.

Likewise, the transport map approximation (Figure 2) occurs in a relatively low-dimensional parameter latent space. It extensively uses highly optimized modern computing kernels such as batch computations, fast sampling of white noise, automatic differentiation, and compile-optimized explicit calculations where all operations are known a priori.

4.3. `LazyDINO` as an amortized inference method

The latent space transport maps can be rapidly constructed during the online phase, making `LazyDINO` a compelling method for real-time inference and a competitive alternative to simulation-based amortized inference (SBAI) methods [87] for solving BIPs with the same PtO map but difference instances of observational data.

Since we create a surrogate for the PtO map, *and not* the likelihood, the cost of `RB-DINO` surrogate construction can be amortized by instancing a new likelihood for any new observational data. In this amortization process (see Figure 3), the construction of the `RB-DINO` surrogate is amortized over many different BIPs defined by the same PtO map and prior. With this in mind, `LazyDINO` is an ideal method for settings where many BIPs are solved for the same system. This class of problems can be found in predictive digital twins, state estimation, and Bayesian optimal experimental design.

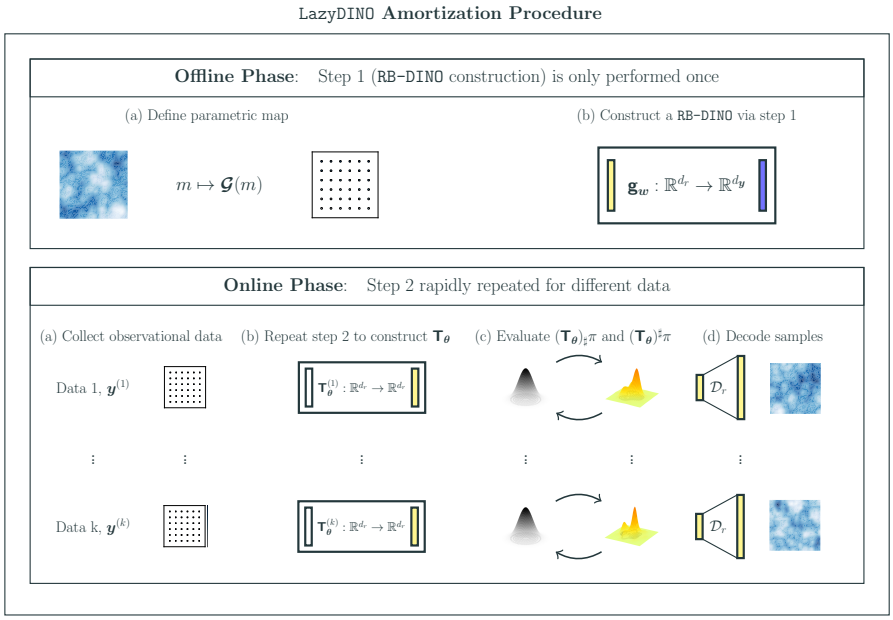

Figure 3: Overview of `LazyDINO` amortization procedure.

Comparison of `LazyDINO` to other SBAI methods. Given joint samples of the latent prior and simulated observational data, the SBVI methods optimize for a *conditional* transport map that matches the pullback distributions, $\boldsymbol{y}^{(j)} \mapsto \mathsf{T}_{\boldsymbol{\theta}}(\boldsymbol{y}^{(j)}, \cdot)^\sharp \pi$, to posteriors at the simulated data samples using an fKL objective. The sample generation and the transport map construction are both performed offline. When the observational data $\boldsymbol{y}^\dagger$ is available, the approximate posterior sampling is performed using inversion at latent prior samples $\mathsf{T}_{\boldsymbol{\theta}}(\boldsymbol{y}^\dagger, \cdot)^{-1}(\boldsymbol{z}^{(j)})$, $\boldsymbol{z}^{(j)} \overset{\text{i.i.d.}}{\sim} \pi$, without the computational bottleneck of model simulations, making it an amortized inference method. Simulating observational data for transport map training in SBAI incurs a similar cost compared to `RB-NO` surrogate construction, requiring PtO map samples with noise perturbation,

i.e., $\boldsymbol{y}^{(j)} := \boldsymbol{\mathcal{G}}(m^{(j)}) + \boldsymbol{n}^{(j)}$ with $m^{(j)} \overset{\text{i.i.d.}}{\sim} \mu$ and $\boldsymbol{n} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \Gamma_n)$. The training also only requires easy-to-evaluate quantities related to the latent prior; see [87].

Despite ostensibly relaxed training requirements, SBAI is much lower in sample efficiency than `LazyDINO`. Given limited PtO map samples, SBAI attempts to directly approximate all posteriors (i.e., posteriors for all instances of observational data), whereas our approach invests these samples in surrogate construction to gain almost unlimited access to all surrogate-approximated posteriors. As a result, our approach leads to a much smaller transport map approximation error than SBAI. Furthermore, due to our efficient `RB-DINO` surrogate construction, the transport map approximation error of SBAI is much higher than the surrogate posterior approximation error in `LazyDINO`, leading to more than two orders of magnitude lower sample efficiency of SBAI observed in our numerical results in Section 6.

All optimization problems of SBAI are solved offline, which is often regarded as an advantage. In contrast, `LazyDINO` requires solving an optimization problem for online posterior sampling at each instance of observational data. However, for popular transport map parametrizations such as conditional normalizing flows, sampling requires solving a root-finding problem in $\mathbb{R}^{d_r}$ for map inversion, which incurs a non-negligible cost in practice. The inversion-to-sample approach of SBAI can be more costly than the optimize-to-sample approach of `LazyDINO` in some situations, e.g., when a large number of approximate posterior samples is needed for each instance of observational data. We provide concrete numerical evidence to support these claims in Section 6.3.

Desired Algorithm Characteristics \ **Posterior estimate**	Ground truth (MCMC)	LA-baseline	`LazyMap`	SBAI	`LazyNO`	`LazyDINO`
Parallel sampling from posterior estimate	✗	✓	✓	✓	✓	✓
Direct sampling, no inversion $\mathbf{T}_{\boldsymbol{\theta}}^{-1}$ required	–	–	✓	✗	✓	✓
Uses a neural surrogate PtO map $\mathbf{g}_w$	✗	✗	✗	✗	✓	✓
Parallelly-sampled training data outputs	–	–	–	$\boldsymbol{y}^{(j)}$	$\boldsymbol{g}^{(j)}$	$\boldsymbol{g}^{(j)}$, $\boldsymbol{J}_r^{(j)}$
Amortizes inversion of posteriors $\mu^{\boldsymbol{y}}$	✗	✗	✗	✓	✗	✗
Uses $D\boldsymbol{\mathcal{G}}$ evaluations	✓	✓	✓	✗	✗	✓
Amortizes evaluation of $\boldsymbol{\mathcal{G}}, D\boldsymbol{\mathcal{G}}$	✗	✗	✗	✓	✓	✓
Embarrassingly parallel $\boldsymbol{\mathcal{G}}$ evaluation	✗	✗	✓	✓	✓	✓

Table 2: **Posterior estimate comparison.** (✓/✗) refer to (true/false). The first two posterior estimates (left two columns) are the ground truth and Laplace approximation baseline (LA-Baseline). The MCMC method we use exhibits none of our desired algorithm characteristics, but its samples serve as a trustworthy ground truth. All competing posterior estimation methods (right four columns) use the same parameter dimension reduction and offer parallel i.i.d. approximate posterior sampling once trained. We use an orange checkmark ✓ to highlight that `LazyMap` can only partially exploit embarrassingly parallel evaluation of the PtO map and its Jacobians, in particular only within each stochastic optimization iteration. We note that SBVI fully amortizes Bayesian inversion, while `LazyNO` and `LazyDINO` only offer amortization of $\boldsymbol{\mathcal{G}}, D\boldsymbol{\mathcal{G}}$ across the estimation of all BIPs.

5. Setup of the numerical studies

In this section, we describe the setup of the numerical examples, including the description of the two BIP examples, neural network architectures, training procedures, and posterior error measures. First, we define the reduced basis dimension, the architecture and training of the `RB-DINO` surrogate, and the architecture and training of the transport map in Section 5.3, and 5.4. Then, we introduce the error measures for operator learning and posterior approximations in Section 5.5. We proceed by first defining the two PDE problems and associated inverse problems. In both cases, we consider nonlinear elliptic PDEs defined on 2D rectangular domains $\Omega \subset \mathbb{R}^2$.

For both problems, we define our Gaussian priors using Matérn covariance operators given by an elliptic operator on $\mathscr{M} \coloneqq L^2(\Omega)$:

$$\mathcal{C} = (-\gamma \nabla \cdot (\boldsymbol{A} \nabla) + \delta \mathrm{Id}_{\mathscr{M}})^{-2}. \tag{44}$$

Here $\gamma, \delta > 0$ are scalar parameters that control the marginal variance and correlation lengths of the random field samples, and $\boldsymbol{A} \in \mathbb{R}^{2\times 2}$ is a symmetric positive definite matrix that induces anisotropy in the random field samples. In both cases, we employ Robin boundary conditions to control boundary artifacts in the samples; see [94, Equation 37] and [95].

In both cases, $\mathscr{M}$ is approximated using linear triangular finite elements, while the state spaces $\mathscr{U} \subset H^1(\Omega;\mathbb{R}), H^1(\Omega;\mathbb{R}^2)$, respectively, are both approximated using quadratic triangular finite elements. In both cases, the reference solution maps utilize Newton–Raphson methods using sparse direct solvers for each Newton iteration. Notably, sparse direct solvers lead to efficient computation of Jacobian training data; see [29, 30] for more information on Jacobian computation.

The PtO map $\boldsymbol{\mathcal{G}}$ is defined by composing the PDE solution operator $\mathcal{F} : \mathscr{M} \to \mathscr{U}$ and an observation operator $\boldsymbol{\mathcal{O}} : \mathscr{U} \to \mathbb{R}^{d_{\boldsymbol{y}}}$. The inverse problems arise from generating four synthetic observations, each obtained by sampling the prior, evaluating the PtO map, and applying additive white noise.

5.1. Example I: Inference of the diffusivity field in a nonlinear reaction–diffusion PDE

For our first example, we consider the following nonlinear reaction–diffusion PDE for $u : \Omega = [0,1]^2 \to \mathbb{R}$:

$$\begin{aligned}
-\nabla_{\boldsymbol{s}} \cdot \exp(m(\boldsymbol{s}))\nabla u(\boldsymbol{s}) + u(\boldsymbol{s})^3 &= 0, \quad && \boldsymbol{s} \in (0,1)^2, && \text{(45a)}\\
\exp(m(\boldsymbol{s}))\nabla u(\boldsymbol{s}) \cdot \boldsymbol{n} &= 0, \quad && \boldsymbol{s} \in \Gamma_{\text{left}} \cup \Gamma_{\text{right}}, && \text{(45b)}\\
u(\boldsymbol{s}) &= 1, \quad && \boldsymbol{s} \in \Gamma_{\text{top}}, && \text{(45c)}\\
u(\boldsymbol{s}) &= 0, \quad && \boldsymbol{s} \in \Gamma_{\text{bottom}}, && \text{(45d)}
\end{aligned}$$

where Γ_{left}, Γ_{right}, Γ_{top}, and Γ_{bottom} denote the left, right, top and bottom boundaries of the unit square, and $\boldsymbol{n}$ is the outward unit normal vector. The inverse problem is to find the log-diffusivity field $m : (0,1)^2 \to \mathbb{R}$ that best matches noisy observations of u at a set of spatial positions.

We use a regular grid with 40×40 cells. The choice of linear triangular finite elements for $\mathscr{M}$ leads to $1,681$ degrees of freedom (DoFs). The choice of quadratic triangular finite elements for $\mathscr{U}$ leads to $3,362$ DoFs. For the prior covariance (44), we choose $\gamma = 0.03$, $\delta = 3.33$, which leads to a point-wise marginal variance around 9 and a spatial correlation length of around 0.1. We take $\boldsymbol{A} = \mathrm{Id}_{\mathbb{R}^2}$. We define the observation operator using $d_{\boldsymbol{y}} = 25$ randomly sampled interior points $\{\boldsymbol{s}_{\text{obs}}^{(j)}\}_{j=1}^{d_{\boldsymbol{y}}}$, i.e., $(\boldsymbol{\mathcal{O}}(u))_j = \int_{\boldsymbol{B}_\epsilon(\boldsymbol{s}_{\text{obs}}^{(j)})} u(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}$, where $\boldsymbol{B}_\epsilon(\boldsymbol{s}) \subset (0,1)^2$ is a ball around $\boldsymbol{s}$ with a small radius $\epsilon > 0$. The noise distribution has covariance $\Gamma_n = 1.94 \times 10^{-3}\mathrm{Id}_{\mathbb{R}^{d_{\boldsymbol{y}}}}$, which corresponds to a signal-to-noise ratio of around 500. We visualize the synthetic data set in Figure 4.

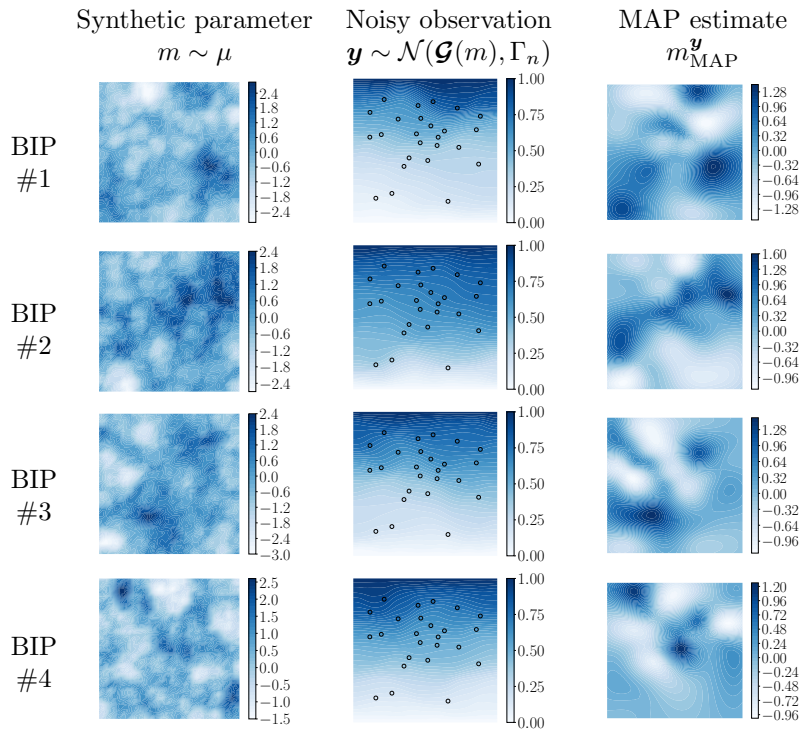

Figure 4: **Example I**. Setup for inferring the diffusivity field in a nonlinear reaction–diffusion PDE detailed in Section 5.1. For each BIP (#1–4), we show the data-generating synthetic parameter m drawn from the prior, the synthetic data $\boldsymbol{y}$ placed on top of the PDE solution at m, and the MAP estimate $m^{\boldsymbol{y}}_{\text{MAP}}$, i.e., the solution of the deterministic inverse problem.

5.2. Example II: Inference of a heterogeneous hyperelastic material property

For our second example, we consider the uniaxial tensile test of a hyperelastic thin film. The inverse problem aims to recover Young's modulus field, which characterizes spatially varying material strength, from measurements of the material deformation. This problem is of interest to the characterization of heterogeneous material properties from deformation data, see for example [96]. Similar Bayesian inverse problems have been considered in [29, 46].

Let $\Omega = (0, 2) \times (0, 1)$ be a normalized material domain. The material coordinates $\boldsymbol{s} \in \Omega$ of the reference configuration are mapped to the spatial coordinates $\boldsymbol{s}+\boldsymbol{u}(\boldsymbol{s})$ of the deformed configuration, where $\boldsymbol{u} : \Omega \to \mathbb{R}^2$ is the material displacement. The strain energy of the hyperelastic material $\mathcal{W}_e$ depends on the deformation gradient, i.e., $\mathcal{W}_e = \mathcal{W}_e(\boldsymbol{F})$ where $\boldsymbol{F} = \text{Id}_{\mathbb{R}^{2\times 2}} + \nabla \boldsymbol{u}$. We consider the neo-Hookean model for the strain energy density:

$$\mathcal{W}_e(\boldsymbol{F}) = \frac{\mu_e}{2}(\text{tr}(\boldsymbol{F}^\top \boldsymbol{F}) - 3) + \frac{\lambda_e}{2}\left(\ln \det(\boldsymbol{F})\right)^2 - \mu_e \ln \det(\boldsymbol{F}). \tag{46}$$

Here, λ_e and μ_e are the Lamé parameters, and they are related to Young's modulus E_Y and Poisson ratio ν_P under the plain strain assumption:

$$\lambda_e = \frac{E_Y \nu_P}{(1+\nu_P)(1-2\nu_P)}, \qquad \mu_e = \frac{E_Y}{2(1+\nu_P)}. \tag{47}$$

We assume $\nu_P = 0.4$, and a spatially-varying normalized Young's modulus, $E_Y : \Omega \to (E_{Y\,\text{min}}, E_{Y\,\text{max}})$ with $0 < E_{Y\,\text{min}} < E_{Y\,\text{max}}$. We represent E_Y through a parameter field $m : \Omega \to \mathbb{R}$ as follows

$$E_Y(m(\boldsymbol{s})) = \frac{1}{2}\left(E_{Y\,\text{max}} - E_{Y\,\text{min}}\right)\left(\text{erf}(m(\boldsymbol{s})) + 1\right) + E_{Y\,\text{min}},$$

where $\mathrm{erf} : \mathbb{R} \to (-1, 1)$ is the error function. We use $E_{Y\,\min} = 1$ and $E_{Y\,\max} = 7$. The first Piola–Kirchhoff stress tensor is given by $\boldsymbol{P}_e(m, \boldsymbol{F}) = 2\partial \mathcal{W}_e(m, \boldsymbol{F})/\partial \boldsymbol{F}$. Assuming a quasi-static model with negligible body forces, the balance of linear momentum leads to the following nonlinear PDE:

$$\begin{aligned} \nabla_{\boldsymbol{s}} \cdot \boldsymbol{P}_e(m, \boldsymbol{F})(\boldsymbol{s}) &= \boldsymbol{0}, && \boldsymbol{s} \in \Omega; && \text{(48a)} \\ \boldsymbol{u}(\boldsymbol{s}) &= \boldsymbol{0}\,, && \boldsymbol{s} \in \Gamma_{\text{left}}; && \text{(48b)} \\ \boldsymbol{u}(\boldsymbol{s}) &= 3/2\,, && \boldsymbol{s} \in \Gamma_{\text{right}}; && \text{(48c)} \\ \boldsymbol{P}_e(m, \boldsymbol{F})(\boldsymbol{s}) \cdot \boldsymbol{n} &= \boldsymbol{0}, && \boldsymbol{s} \in \Gamma_{\text{top}} \cup \Gamma_{\text{bottom}}; && \text{(48d)} \end{aligned}$$

where Γ_{top}, Γ_{right}, Γ_{bottom}, and Γ_{left} denote the material domain's top, right, bottom, and left boundary.

We use a regular grid with 64×32 cells. The choice of linear triangular finite elements for $\mathscr{M}$ leads to $2,145$ DoFs. The choice of quadratic triangular vector finite elements for $\mathscr{U}$ leads to $16,770$ DoFs. The Newton–Raphson method is initialized with the homogenous deformation field. For the prior covariance (44), we induce spatial anisotropy via the following matrix

$$\boldsymbol{A} = \begin{bmatrix} \theta_1 \sin(\alpha)^2 + \theta_2 \cos(\alpha)^2 & (\theta_1 - \theta_2)\sin(\alpha)\cos(\alpha) \\ (\theta_1 - \theta_2)\sin(\alpha)\cos(\alpha) & \theta_1 \cos(\alpha)^2 + \theta_2 \sin(\alpha)^2 \end{bmatrix},$$

where $\theta_1 = 2$ and $\theta_2 = 1/2$, $\alpha = \arctan(2)$. Additionally, we choose $\gamma = 0.3$ and $\delta = 3.3$ which leads to a point-wise marginal variance around 1, and a spatial correlation of around 2 and 0.5 respectively, perpendicular to and along the left bottom to top right diagonal of the material domain. We define the observation operator using 32 equally spaced interior points $\{\boldsymbol{s}_{\text{obs}}^{(j)}\}_{j=1}^{32}$, similar to Example I. This leads to $d_{\boldsymbol{y}} = 64$. The noise distribution has covariance $\Gamma_n = 2.86 \times 10^{-3} \mathrm{Id}_{d_{\boldsymbol{y}}}$, which corresponds to a signal-to-noise ratio of around 500. We visualize the synthetic data set in Figure 5.

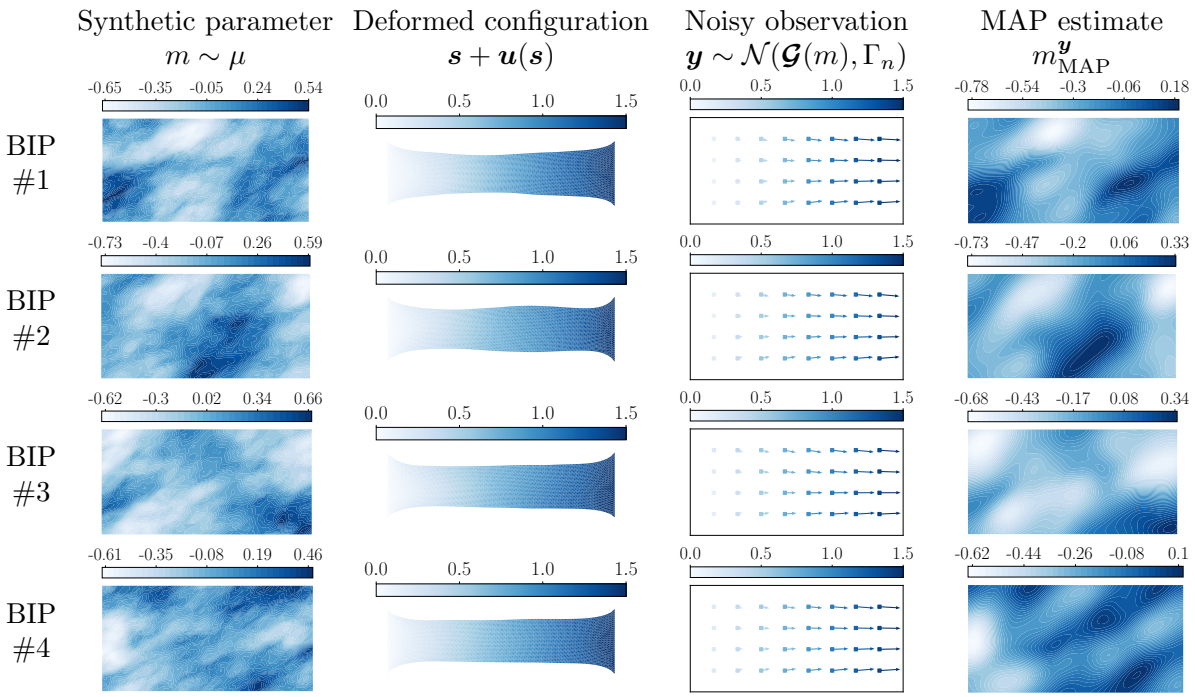

Figure 5: **Example II.** Setup for inferring a heterogeneous hyperelastic material property detailed in Section 5.2. For each BIP (#1–4), we visualize the synthetic parameter m drawn from the prior, the corresponding deformed configuration, the synthetic displacement data $\boldsymbol{y}$, and the MAP estimate $m^{\boldsymbol{y}}_{\text{MAP}}$, i.e., the solution of the deterministic inverse problem.

5.3. Surrogate architecture and training

Reduced basis. For both problems, we chose the dimension of the parameter latent space to be $d_r = 200$ and used 1000 MC samples to compute the reduced basis. We fixed the reduced basis dimension for all studied variational inference methods. Our numerical examples are focused on comparing our proposed `LazyDINO` with other variational inference methods. Therefore, the effects of varying d_r and MC sample sizes for reduced basis construction are not studied in this work. The eigenvalue decay and basis functions for both examples are visualized in Appendix J.

Neural network architecture and training. For both examples, we choose a dense multi-layer perceptron (MLP) as $\mathbf{g}_w$ with 7 hidden layers, each with a width of 400 and a Gaussian error linear unit (GELU) activation function. We train $\mathbf{g}_w$ as described in Algorithm 2 using Adam with $1,500$ epochs and a batch size 25. For `RB-DINO` training, we used a learning rate of 1×10^{-3} and decreased the learning rate to 3×10^{-4} for the final 375 epochs. We found that many other training tricks, such as batch normalization or learning rate decay scheduling, were not necessary to produce good generalization for `RB-DINO`.

Remark 3. *To maintain training stability and prevent overfitting for conventional L^2_μ training, we needed to decrease the learning rate and modify the number of epochs, depending on the training data size. Details reported in Appendix I.*

Implementations of the neural networks and training procedures can be found in `dinox`, a `JAX` derivative-informed neural operators library.

5.4. Transport map architecture and training

Architecture. We chose inverse autoregressive flow (IAF) [91] as our the transport map $\mathbf{T}_{\boldsymbol{\theta}} : \mathbb{R}^{d_r} \to \mathbb{R}^{d_r}$, with $N_{\mathbf{T}} = 30$ transport map layers $\tau_i : \mathbb{R}^{d_r} \to \mathbb{R}^{d_r}$ and random input permutation, i.e. $\mathbf{T}_{\boldsymbol{\theta}} = \mathbf{T}_{\boldsymbol{\theta}_{N_{\mathbf{T}}}} \circ \cdots \circ \mathbf{T}_{\boldsymbol{\theta}_1}$ with $\mathbf{T}_{\boldsymbol{\theta}_i} = \boldsymbol{\tau}_{\boldsymbol{\theta}_i} \circ \boldsymbol{P}_i$ where $\boldsymbol{P}_i : \mathbb{R}^{d_r} \to \mathbb{R}^{d_r}$ is the random permutation. Each transport map $\boldsymbol{\tau}_{\boldsymbol{\theta}_i}$ is an autoregressive MLP, where inputs are Boolean masked to ensure triangular dependence, i.e., $\boldsymbol{\tau}_{\boldsymbol{\theta}_i}(\boldsymbol{z}_j) = \boldsymbol{\tau}_{\boldsymbol{\theta}_i}(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{j-1})$. Each autoregressive MLP has 4 hidden layers, each with a width of 400 and GELU activation. In total, the trained parameters are $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{N_{\mathbf{T}}})$. For SBAI, the conditional normalizing flow is constructed with a transport map $\mathbf{T}_{\boldsymbol{\theta}} : \mathbb{R}^{d_r} \times \mathbb{R}^{\mathbb{R}^{d_y}}$ with a masked autoregressive flow (MAF) with the same architecture, but with a larger input dimension to account for conditioning on observations. We also use the tanh activation function since it empirically produced more stable results. All TMVI methods are implemented and trained via `lazydinox`, a `JAX` library for `LazyDINO` algorithms.

Training. The training procedure for all compared methods are taken to be as similar as possible. For `LazyDINO` and `LazyNO`, we train $\mathbf{T}_{\boldsymbol{\theta}}$ using Adamax [97] with 5 batch sizes, as defined in Algorithm 3. For minimization j, we use I_j iterations using a B_j–sample MC gradient estimator and learning rate α_j, labeled here as (I_j, B_k, α_j): $\Big\{(5k, 200, 5 \times 10^{-3}), (1k, 500, 5 \times 10^{-3}), (1k, 2,000, 5 \times 10^{-3}), (1k, 5,000, 5 \times 10^{-3}), (1k, 7,500, 5 \times 10^{-4})\Big\}$, where we decrease the learning rate slightly when we reach the final stochastic approximation batch sample size $7,500$. In contrast, for `LazyMap`, since each Adamax iteration involves PtO map evaluations (referred to as training samples in our results), we use only one batch size, $(I_0, B_0, \alpha_0) = (200, 640, 5 \times 10^{-3})$ for Example I and $(I_0, B_0, \alpha_0) = (200, 100, 5 \times 10^{-3})$ for Example II, for a total of 128,000 and 20,000 PtO map evaluations, respectively. For comparison in the proceeding section, error measures are recorded after steps $\#(5, 10, \ldots, 320, 640)$, which equal training sample sets of size $(1,000, 2,000, \ldots, 64,000, 128,000)$, respectively. For SBAI, we train with batches of size 100 sampled-without-replacement over epochs with a fixed learning rate of 5×10^{-4}. We terminated optimization when the validation error had not decreased in 10 epochs.

5.5. Error measures

Surrogate approximation error. The approximation error of the neural ridge function surrogate $\widetilde{\mathcal{G}}_{\boldsymbol{w}} \circ \mathcal{P}$ for the PtO map approximation, $\boldsymbol{E}_{\mathbf{g}}$, and the PtO map latent Jacobian approximation, $\boldsymbol{E}_{\nabla \mathbf{g}}$, are defined as follows.

$$\boldsymbol{E}_{\mathbf{g}} = \sqrt{\frac{1}{N_{\mathrm{MC}}} \sum_{j=1}^{N_{\mathrm{MC}}} \left[\frac{\|\mathbf{g}_w(\boldsymbol{z}^{(j)}) - \boldsymbol{g}^{(j)}\|^2}{\|\boldsymbol{g}^{(j)}\|^2} \right]} \qquad \text{(Relative PtO map error)}$$

$$\boldsymbol{E}_{\nabla \mathbf{g}} = \sqrt{\frac{1}{N_{\mathrm{MC}}} \sum_{j=1}^{N_{\mathrm{MC}}} \left[\frac{\|\boldsymbol{J}_r^{(j)} - \nabla \mathbf{g}_w(\boldsymbol{z}^{(j)})\|_F^2}{\|\boldsymbol{J}_r^{(j)}\|_F^2} \right]} \qquad \text{(Relative latent Jacobian error)}$$

We compute the errors by using $N_{\mathrm{MC}} = 5,000$ joint samples of the prior, whitened PtO evaluations and its latent Jacobian evaluations.

Posterior approximation error. Posterior approximation accuracy can be assessed in many ways. It is important to ensure posterior accuracy where probability mass is more present, i.e., measures of central concentration or tendency, such as central moments or modes. Accuracy can also be assessed via probability divergences, which measure the overall deviation from the posterior. Accounting for these various forms of accuracy measurement, we consider the quality of posterior approximation under a nonlinear transport map $\mathcal{T}$ via *moment discrepancies* and *density-based diagnostics*, which are described as follows.

1. **Moment discrepancies.** Let $\overline{m}^{\boldsymbol{y}} \in \mathscr{M}$, $\mathcal{C}^{\boldsymbol{y}} \in \mathrm{HS}(\mathscr{M})$ denotes the mean and covariance of $\mu^{\boldsymbol{y}}$ and $\mathcal{S}_{25}^{\boldsymbol{y}} \in \mathbb{R}^{25\times 25\times 25}$ denotes the skewness of $\mu^{\boldsymbol{y}}$ in the leading 25 latent space coordinates. Let $\overline{m}^{\mathcal{T}}$, $\mathcal{C}^{\mathcal{T}}$, and $\mathcal{S}_{25}^{\mathcal{T}}$ denote the same quantities for $\mathcal{T}_\sharp \mu$. We consider the following relative error in the moments

$$\boldsymbol{E}_{\mathrm{mean}} = \left\|\overline{m}^{\boldsymbol{y}} - \overline{m}^{\mathcal{T}}\right\|_{\mathscr{M}} / \left\|\overline{m}^{\boldsymbol{y}}\right\|_{\mathscr{M}} \qquad \text{(Relative mean error)}$$

$$\boldsymbol{E}_{\mathrm{cov}} = \left\|\mathcal{C}^{\boldsymbol{y}} - \mathcal{C}^{\mathcal{T}}\right\|_{\mathrm{HS}(\mathscr{M})} / \left\|\mathcal{C}^{\boldsymbol{y}}\right\|_{\mathrm{HS}(\mathscr{M})} \qquad \text{(Relative covariance error)}$$

$$\boldsymbol{E}_{\mathrm{skew}} = \left\|\mathcal{S}_{25}^{\boldsymbol{y}} - \mathcal{S}_{25}^{\mathcal{T}}\right\|_F / \left\|\mathcal{S}_{25}^{\boldsymbol{y}}\right\|_F \qquad \text{(Relative skewness error)}$$

   The central moments of both $\mu^{\boldsymbol{y}}$ and $\mathcal{T}_\sharp \mu$ are estimated using samples, where samples from $\mu^{\boldsymbol{y}}$ are obtained using up to 5×10^6 MCMC samples using a simplified manifold MCMC method [29, 98]. The discrepancies are reported in terms of percentages.

   Since all central moments must converge as a posterior estimator converges to the posterior, analyzing moment discrepancies of varying orders together is more helpful than analyzing them independently. Estimating higher-order statistics becomes progressively more challenging, so we consider only the first three moments and compute the skewness in the leading dimensions of the latent space.

2. **Density-based diagnostics.** Let $\Phi_{\mathcal{T}}(m) \coloneqq \log\left(\frac{\mathrm{d}\mu}{\mathrm{d}(\mathcal{T}_\sharp \mu)}(m)\right)$ denote the log density of the prior with respect to the pushforward distribution; see Appendix H for explicit forms. Let $\widetilde{w}(m) = \exp(-2\Phi^{\boldsymbol{y}}(m) + 2\Phi_{\mathcal{T}}(m))$ denote the unnormalized density of the posterior with respect to the pushforward density and $w(m)$ denote its normalization by $\mathbb{E}_{m\sim\mathcal{T}_\sharp\mu}[\widetilde{w}(m)]$. We consider the following quantities related to the quality of each posterior approximation:

$$\boldsymbol{E}_{\mathrm{fKL}} = \mathbb{E}_{m\sim\mathcal{T}_\sharp\mu}\left[\Phi^{\boldsymbol{y}}(m) - \Phi_{\mathcal{T}}(m)\right] + C_1 \qquad \text{(Shifted rKL divergence)}$$

$$\boldsymbol{E}_{\mathrm{rKL}} = \mathbb{E}_{m\sim\mathcal{T}_\sharp\mu}\left[w(m)(-\Phi^{\boldsymbol{y}}(m) + \Phi_{\mathcal{T}}(m))\right] + C_2 \qquad \text{(Shifted ANIS fKL divergence)}$$

$$\mathrm{ESS}_N\% = \frac{\left(\frac{1}{N}\sum_{j=1}^N \widetilde{w}(m^{(j)})\right)^2}{\frac{1}{N}\sum_{j=1}^N \widetilde{w}(m^{(j)})^2} \times \frac{100\%}{N}, \quad m^{(j)} \overset{\text{i.i.d.}}{\sim} \mathcal{T}_\sharp\mu \qquad \text{(ANIS effective sample size percentage)}$$

$$\boldsymbol{E}_{\mathrm{MAP}} = \left\|m^{\boldsymbol{y}}_{\mathrm{MAP}} - m^{\mathcal{T}_{\mathrm{MAP}}}\right\|_{\mathscr{M}} / \left\|m^{\boldsymbol{y}}_{\mathrm{MAP}}\right\|_{\mathscr{M}} \qquad \text{(Relative MAP point error)}$$

We use both rKL and fKL to measure the posterior approximation error. The former measures the optimality gap due to surrogate error in `LazyNO` and `LazyDINO`. On the other hand, rKL can be small when the pushforward distribution is overly concentrated. Therefore, considering both fKL and rKL together provides a fuller picture of posterior approximation accuracy. We use the auto-normalized importance sampling weights [99] to compute a biased but consistent estimator for the fKL. We note that rKL is shifted by the normalization constant, and both rKL and fKL are additionally shifted by a constant for visualizations in the log scale. We use 10^5 i.i.d. samples from the pushforward to estimate the expectations in the rKL and fKL divergences.

We also consider the effective sample size percentage estimated using N i.i.d. samples from the pushforward distribution, denoted $\mathrm{ESS}_N\%$. We take $N = 10^5$ in our numerical studies. Effective sample size (ESS) is a commonly used diagnostic to assess the quality of approximate posterior sampling, and it has been observed that producing large effective sample percentages can be difficult for many numerical methods [100, 101]. It is related to the forward χ^2 divergence $\chi^2(\mu^{\boldsymbol{y}}||\mathcal{T}_\sharp\mu) \approx 1 - \mathrm{ESS}_N\%/100$ [102].

Lastly, we study convergence in the MAP estimate, where the ground truth MAP point is obtained by LA.

6. Numerical results

In this section, we present numerical results for the two problems described in the previous section. First we show the generalization errors accomplished when training `RB-DINO` and `RB-NO` ridge function surrogates with different training sample sizes in Section 6.1. These errors are tied to the accuracy of the surrogate-based posterior approximation via Theorem 3.1, and to the optimality gap in surrogate-driven LMVI via Corollary 3.3.

In the subsequent results, we compare the posterior errors for `LazyDINO` and `LazyNO`. As points of comparison, we additionally consider the Laplace approximation as a baseline, SBAI, as well as `LazyMap`, which utilizes the true PtO map in training instead of the surrogate.

Remark 4. *In the neural operator training results, we compare the computational costs measured in terms of nonlinear PDE solves. The additional costs associated with the `RB-DINO` training are measured in a cost basis that is relative to the nonlinear PDE solves. Since these additional computational costs associated with the latent Jacobian computations are negligible due to the use of sparse direct solvers [29, Section 4.3], this point of comparison is not considered for the posterior approximation error comparisons.*

6.1. Neural operator ridge function generalization

We begin by assessing the results of `RB-DINO`, and `RB-NO` surrogate construction. The results for Example I (a nonlinear reaction–diffusion PDE) can be found in Figure 6, while the results for Example II (deformation of a hyperelastic thin film) can be found in Figure 7. Overall, the trend demonstrates that the derivative-informed learning method leads to a significant cost reduction in learning the latent representation of the PtO map and its Jacobian compared to conventional supervised learning. These results are consistent with those in [29, 30]. We expect that the improvement of the PtO map approximation leads to better fidelity in the surrogate approximated posterior through Theorem 3.1, and the combination of improved PtO map approximation and Jacobian approximation leads to more accurate LMVI optimization through Corollary 3.3.

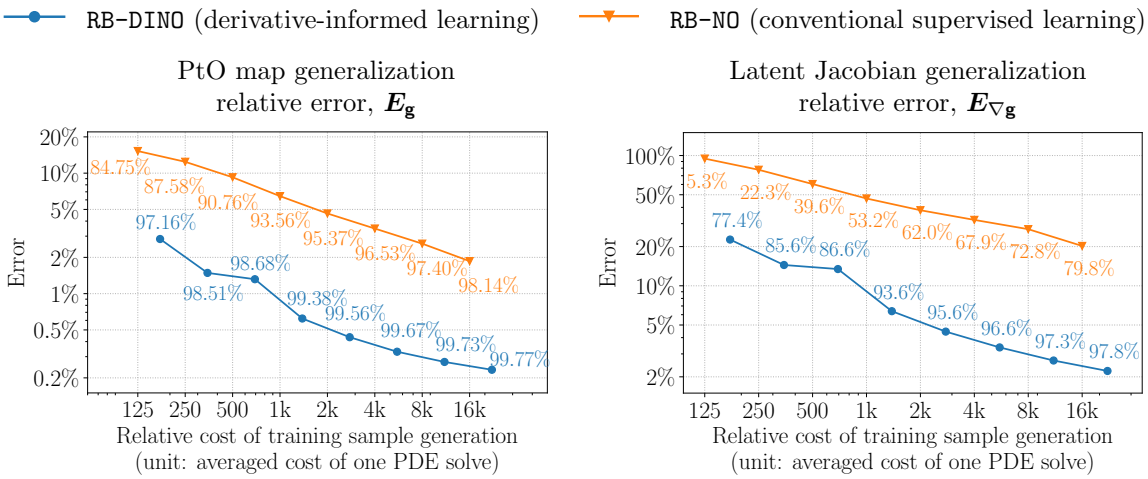

Figure 6: **Example I neural ridge function surrogate testing.** Percentage accuracy, 100% × (1 − error), is also overlaid. Overall, these results demonstrate a significant cost reduction for achieving any given generalization accuracy in both the PtO map and the latent Jacobian via the derivative-informed learning method. While convergence rates are similar, `RB-DINO` has a greater than 64× higher cost efficiency measure in relative errors. Due to the bounds in Section 3.4, we expect this to reflect in increased sample efficiency of `LazyDINO` over `LazyNO` in posterior error measures. We note a statistical anomaly in `RB-DINO` training encountered at 500 training samples, which poses a downstream impact on the posterior error measures in subsequent figures.

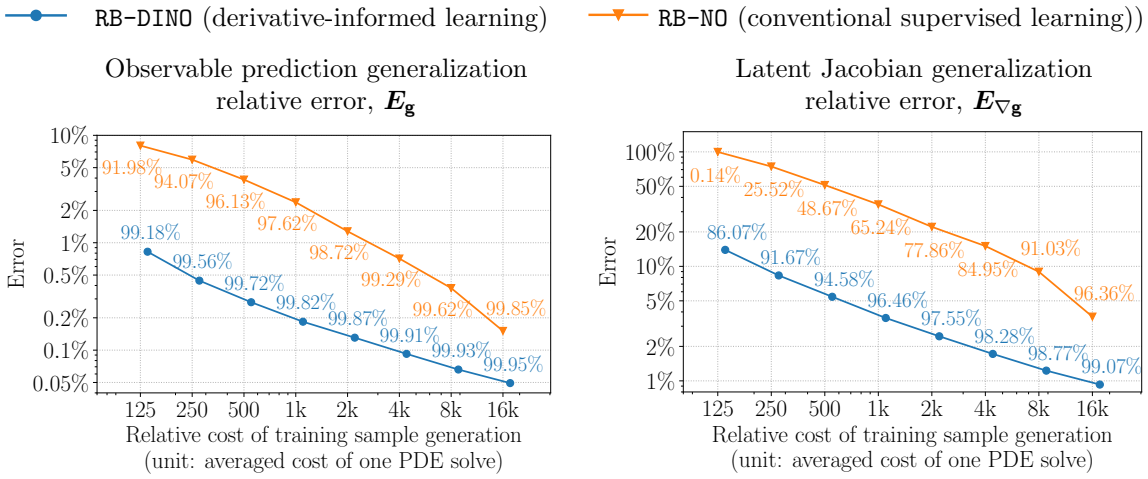

Figure 7: **Example II neural operator ridge function generalization.** In a similar pattern as Figure 6, `RB-DINO` enjoys 8–32× lower data generation cost over `RB-NO` for achieving a given accuracy. We expect this to reflect in increased sample efficiency of `LazyDINO` over `LazyNO` in posterior error measures.

6.2. Posterior approximation error

In this section we investigate the posterior error metrics described in Section 5.5. As a reminder, we compare `LazyDINO` against `LazyNO`, the PDE-driven `LazyMap` as well as two additional baselines: the Laplace approximation and SBAI via conditional transport.

For all plots in this section, the markers for BIP #1–4 are labeled as $\curlyvee$, $\curlywedge$, $\prec$, and $\succ$. The average error is plotted in a darker color, and a line is drawn between each average to visualize the trend. We cut off vertical-axis errors at 200–300%, depending on the plot, for readability since the scale of the errors across methods varies widely.

For the LA-baseline, horizontal lines are included to facilitate visual comparison, even though it is only computed once for each BIP–the horizontal axis, i.e., the number of training samples is not meaningful in this context. A conservative estimate of training cost equivalent to 100 training samples, used for MAP estimation and Hessian-inverse covariance estimation, is marked in the plots. To potentially outperform the LA-baseline in amortized Bayesian inversion, a method should achieve lower posterior error as few amortized PtO and Jacobian evaluations as possible.

We begin by comparing moment discrepancies for Example I and II in Figure 8 and Figure 9, respectively. In general the LA-baseline provided a reasonable baseline point of comparison; and in the case of covariance approximations for Example I, it consistently outperformed each method. In all other cases, however, `LazyDINO` eventually produced substantially better predictions of moments, particularly given a lot of data (e.g., consider the mean error for Figure 9). Of the remaining methods, `LazyDINO` produced the best matching moments for each problem in each training sample size. `LazyNO` was typically the next best performing method, although in some cases SBAI or `LazyMap` performed comparably or slightly better. A notable phenomenon was the relatively poor performance of SBAI, which was typically more than an order of magnitude worse than `LazyDINO`. The poor performance relatively of `LazyMap` is easily explained by the sample intensity required to minimize the latent space transport training objective reliably. In particular, we artificially truncated the sampling budget at $128,000$, while the `LazyDINO/NO` required 16 million total samples over all iterations. This point of comparison demonstrates the essential benefit of the `LazyDINO` approach: by first building a reliable PtO surrogate over the prior using a fixed number of samples, we can later enable an optimization algorithm requiring orders of magnitude more samples.

In the next set of results, we consider various density-based diagnostic criteria, which are defined in Section 5.5. In Figure 10 and Figure 11, we compare the performance of the different methods through their shifted rKL and fKL, ANIS effective sample percentage, and MAP point estimates. As with the moment discrepancy results, we see again the consistent superior performance of `LazyDINO` compared to the other TMVI methods and the LA-baseline for $> 500/1000$ samples for Example I and II, respectively. Notably, the `LazyDINO` effective sample size is orders of magnitude higher than the other methods for both examples in the largest sample case.

6.3. Timing comparisons

In the previous section, we compared various methods on a sample-cost basis. In this section, we consider additional computational speedups, such as parallelism, to make a comparison on a time-cost basis.

We begin by making empirical comparisons of `LazyDINO` and the original `LazyMap` algorithm, taking into account that `LazyMap` can be made more efficient with parallelism. We continue by comparing the online evaluation costs for SBAI and `LazyDINO/NO`, demonstrating that in addition to being much more accurate than SBAI, `LazyDINO/NO` have a smaller overall cost in an amortized setting.

Remark 5. *Compute times will vary based on computing environments, but we note that all GPU computations were performed on Nvidia A100 GPUs with 40 and 80GB of RAM, and all CPU computations were performed on an Intel Xeon Gold 6248R 3.00GHz CPU with 1.2 Terabytes of RAM (CPU computations were compute-bound, not memory bound).*

`LazyMap` vs. `LazyDINO`. The efficiency of `LazyDINO` is impacted by the compute time for the offline phase. Since `LazyMap` can exploit parallelism within each iteration, depending on the parallel computing resources available, `LazyMap` might still be competitive in particular sample size regimes since it does not require first

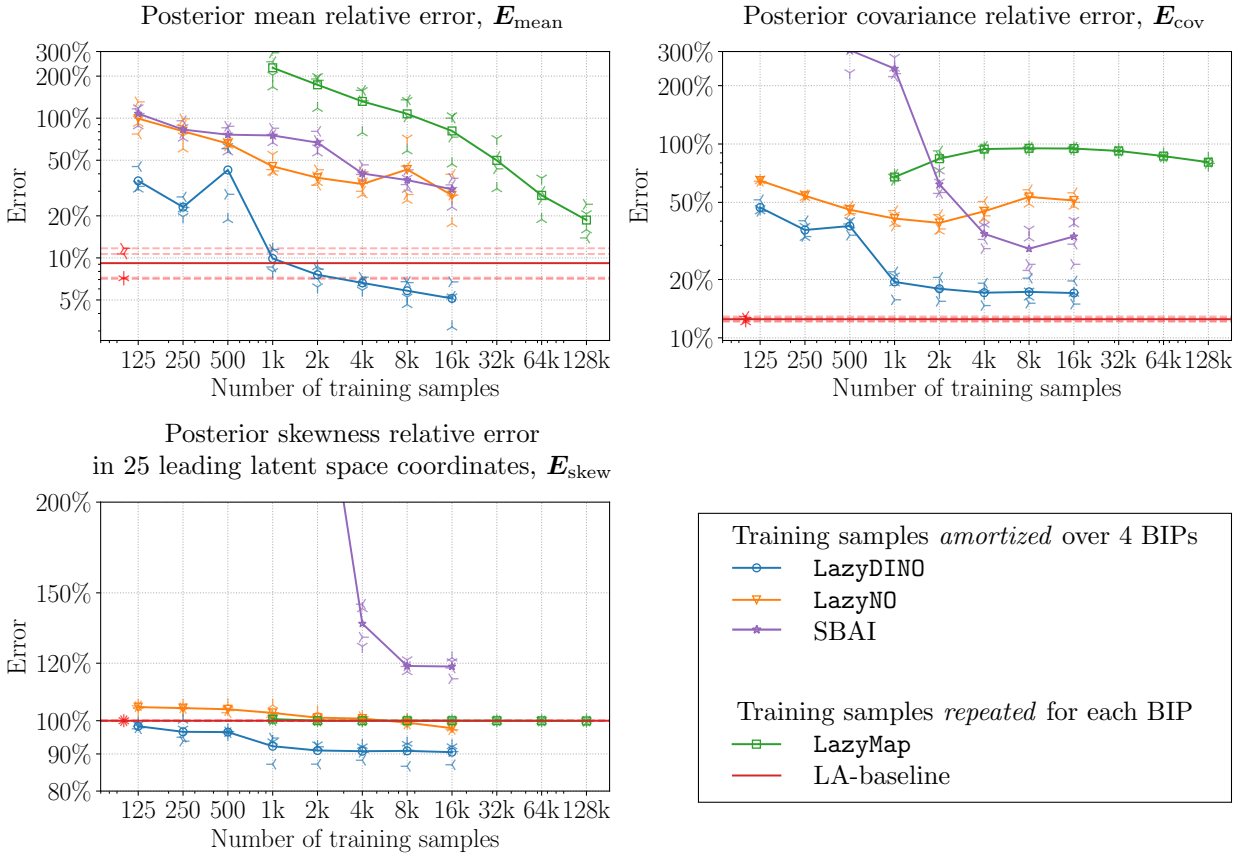

Figure 8: **Example I moment discrepancies.** In all cases, a lower error indicates a better posterior approximation. (`LazyDINO` vs. `LazyNO`): Apart from the statistical anomaly at 500 samples attributed to the stochasticity in surrogate training seen in Figure 6, `LazyDINO` is 64× more sample-efficient measured in mean relative error, over 4× in covariance relative error, and over 64× in skewness relative error. The discrepancy in efficiency is even more pronounced in the higher sample regime ($>$ 500 samples). (`LazyDINO` vs. SBAI): `LazyDINO` is 64× more sample-efficient measured in mean relative error, and SBAI is uncompetitive in covariance and skewness error in all sample regimes. (`LazyDINO` vs. `LazyMap`): In all error measures, `LazyDINO` achieves orders of magnitude higher sample-efficiency compared to `LazyMap`. We also note that since `LazyMap` repeats computations of the PtO for each BIP, the number of training samples in total is 4× the number for the other approaches. (`LazyDINO` vs. LA-baseline): `LazyDINO` achieves lower error in the mean and the skewness, particularly noticeable in the higher sample size regime. While no approach studied in this work can achieve lower relative error in the covariance compared to LA-baseline, we note that the density-based diagnostics in Figure 10 lend further support to the proposition that `LazyDINO` improves upon the baseline, even for small sample sizes.

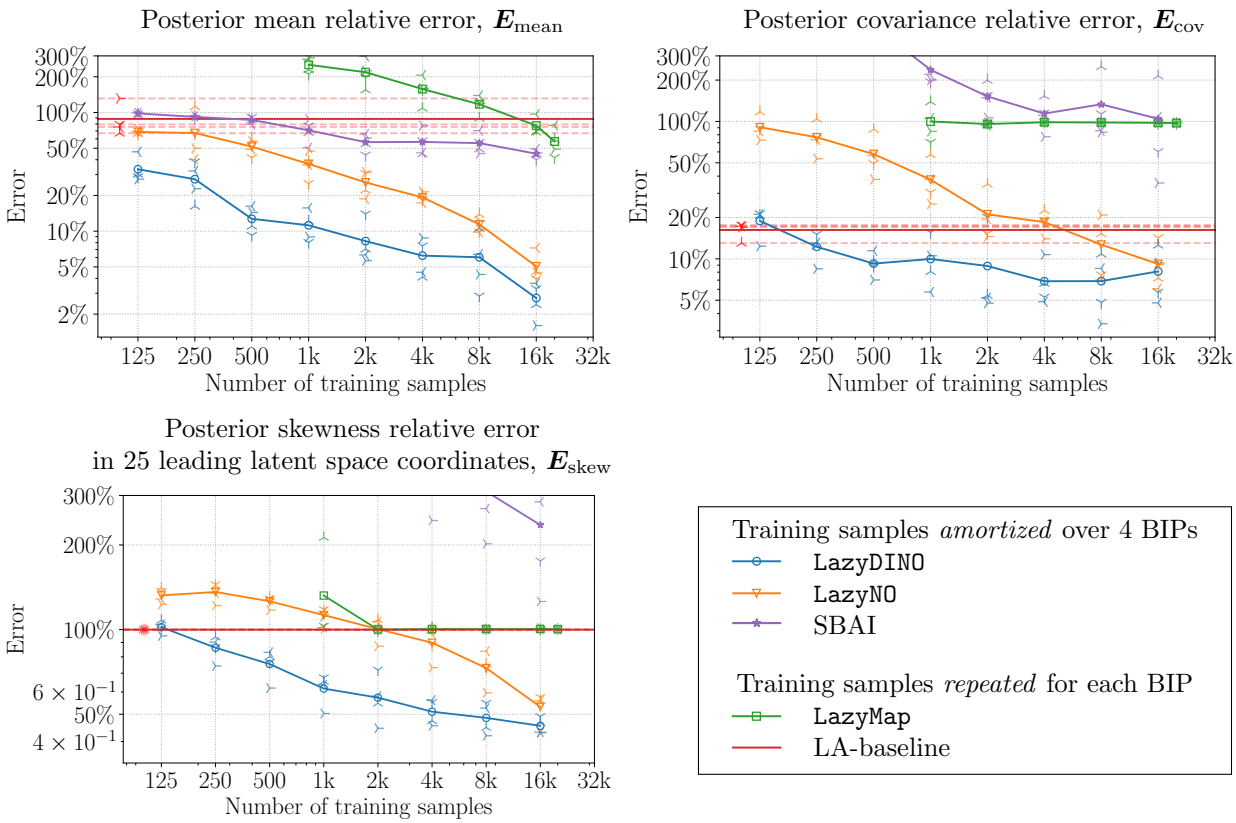

Figure 9: **Example II moment discrepancies.** We observe similar trends as in Example I (Figure 8). (`LazyDINO` vs. `LazyNO`) The trend of consistent outperformance of `LazyNO` is clear in this example; the derivative-informed learning of `RB-DINO` yields $2-16\times$ higher sample efficiency. (`LazyDINO` vs. SBAI) The best-performing SBAI at the high sample regime is still less accurate in posterior approximation compared to the worst `LazyDINO` at the low sample regime. (`LazyDINO` vs. `LazyMap`) `LazyMap` exhausts the training sample budget before performing comparably to `LazyDINO`. (`LazyDINO` vs. `LA-baseline`) At 250 training samples, `LazyDINO` produces lower error than the LA-baseline in all moment discrepancies.

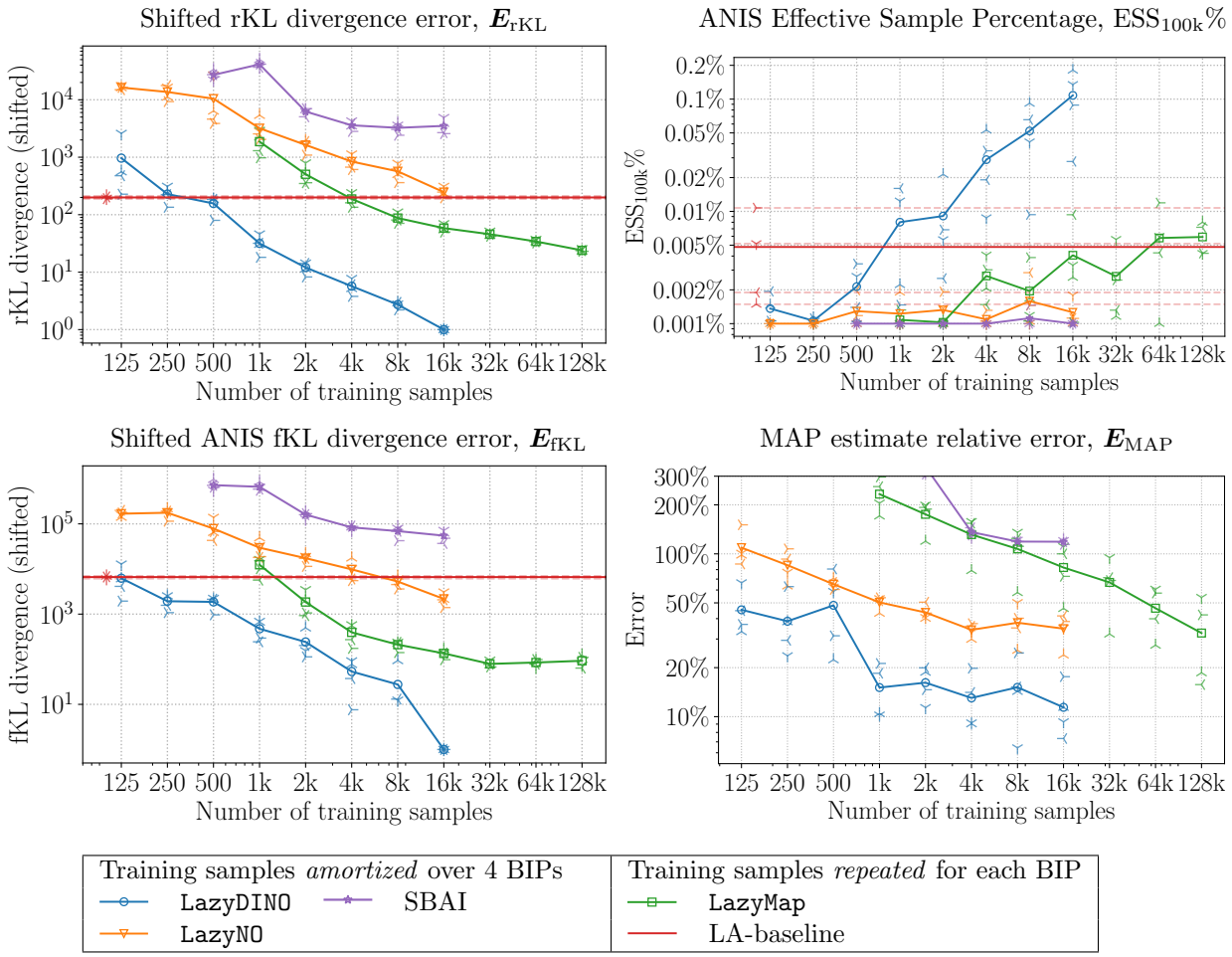

Figure 10: **Example I density-based diagnostics.** Higher values of $\mathrm{ESS}_{100\mathrm{K}}\%$ and lower values of all other diagnostics imply better posterior approximation. We observe similar trends to those observed in the moment discrepancy comparisons in Figure 8. Notably, the `LazyDINO` eventually yields the best error in each case. `LazyDINO` enjoys over 8–128× higher sample efficiency compared to the other methods. Though $\mathrm{ESS}_{100\mathrm{K}}\%$ is low across the board, `LazyDINO` produces an impressive ≈ 100 effective samples while other methods only achieve $\approx$ 1–6 effective samples.

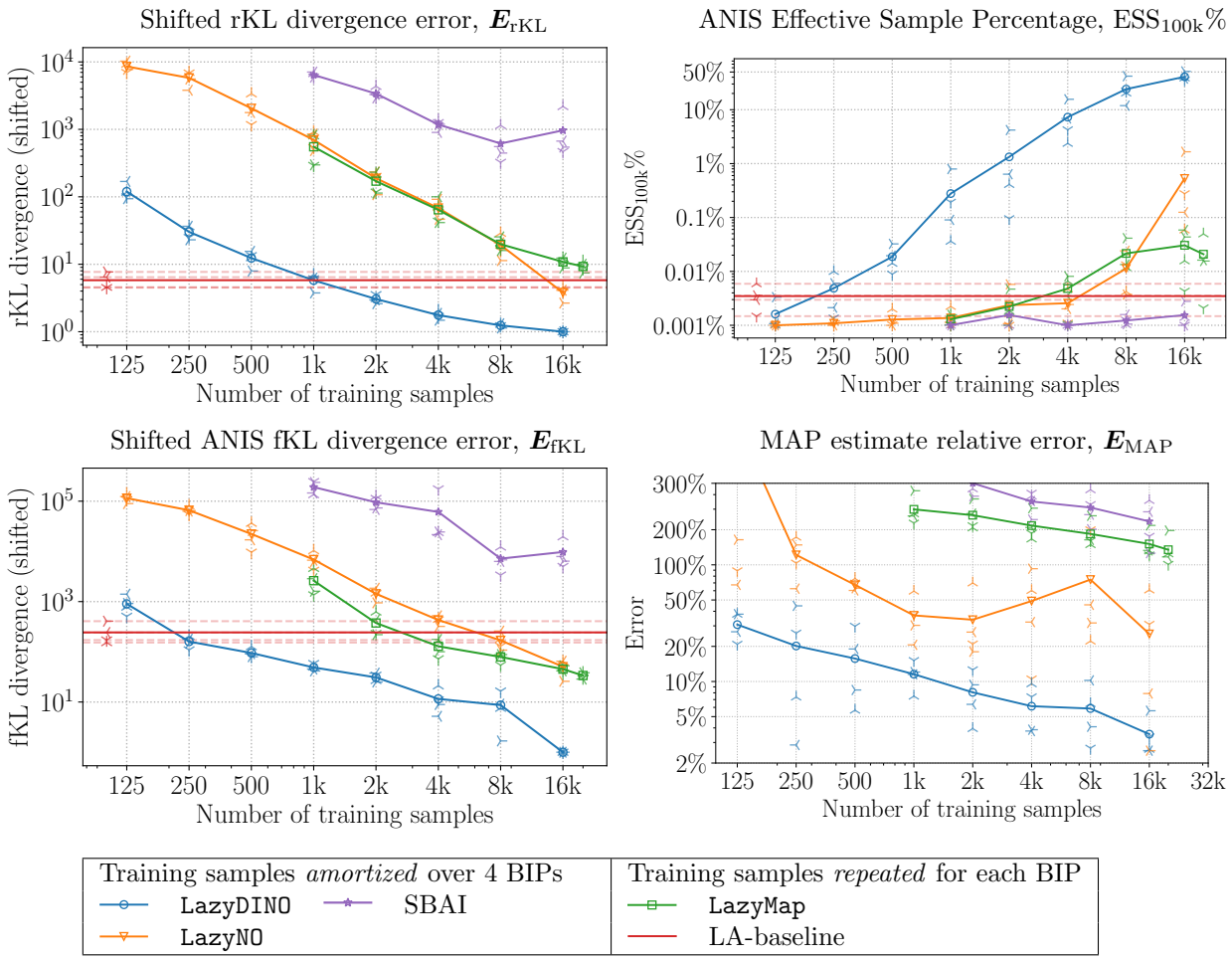

Figure 11: **Example II density-based diagnostics.** We observe similar trends as in Example I. Notably, `LazyDINO` achieves nearly 50% ANIS effective sample percentage as a 100,000-sample independent sampler with 16,000 training samples, 50 to 50,000 times the percentage achieved for competing methods.

training a surrogate. For this comparison, we investigate the relative performance of `LazyMap` and `LazyDINO` for a similar end-to-end computational budget while allowing for parallelism in the `LazyMap` calculations.

In Table 3 and Table 4, we provide a comparison of solution time and posterior mean accuracy for the two methods, given access to 20 concurrent CPU evaluations of the PtO map and Jacobian action. All parallel times provided are theoretical (without accounting for communication), computed to two decimals of accuracy, since we lacked a parallel implementation of the PtO map. We provide actual compute times for training sample data generation for `LazyDINO` and the training times for `LazyDINO` and `LazyMap`. The `LazyDINO` offline phase times are reported amortized across four instances of observation data. All computations repeated for each instance of observation data are reported as an average over the four instances and rounded up to the nearest 10 seconds.

	Category (unit)	**LMVI Method**			
		`LazyMap` (16k)	`LazyMap` (128k)	`LazyDINO` (1k)	`LazyDINO` (16k)
Algorithm steps	Amortized PtO evaluations (sec)	–	–	130/4	1,950/4
	Parallel amortized PtO evaluations (sec)	–	–	6.5 /4	97.5/4
	Amortized Jacobian (sec)	–	–	50/4	750/4
	Parallel amortized Jacobian (sec)	–	–	2.5/4	37.5/4
	Amortized DINO training (sec)	–	–	80/4	1,220 /4
	TMVI training* (sec)	2,710	21,560	460	460
	Parallel TMVI training (sec)	135.5	1078	–	–
Total	Time per BIP (sec)	2,710	21,560	525	1,440
	Parallel time per BIP (sec)	135.5	1078	482.45	798.75
	Relative mean error achieved (%)	80	20	10	5

Table 3: **Example I: `LazyDINO`/`LazyMap` timing comparison.** We include two sample sizes (16k and 128k) for `LazyMap` for comparison with `LazyDINO` at 1k and 16k training data. The total sequential execution times are similar for `LazyMap` (16k) and `LazyDINO` (16k), and the total 20-way parallel execution times are similar for `LazyMap` (128k) and `LazyDINO` (16k). In both cases, the relative mean error achieved is much lower for `LazyDINO`. Moreover, `LazyDINO` (1k) achieves smaller relative mean error than `LazyDINO` (128k) in less time. * denotes the fact that `LazyDINO` already performs batch-vectorized computation, so that parallel computation of the surrogate PtO map is not applicable.

Overall, these results demonstrate that for similar end-to-end computational costs, `LazyDINO` still performs substantially better than `LazyMap`, even allowing 20-way parallelism. We additionally note that, while we considered only relative mean error in the tables, `LazyDINO`'s efficiency gains in other error measures are even higher, as visible in the figures in Section 6.2.

Each iteration of LazyMap is computed with a 200-sample MC gradient estimator, `LazyMap` (16k) refers to 80 stochastic iterations, and `LazyMap` (128k) refers to 640 stochastic iterations. In contrast, since each iteration of `LazyDINO` LMVI is cheap, the training time reported is for the 16 million surrogate PtO and Jacobian actions resulting from the increasing sample size strategy provided in Section 5.4. This demonstrates a key takeaway: for extremely query-intensive algorithms such as the expected risk minimization problem arising in transport map training, surrogates are necessary, and since the associated training costs with the high-fidelity model would be so expensive, one can invest significant offline computations and still save orders of magnitude in computational costs.

	Category (unit)	Method			
		LazyMap (1k)	LazyMap (16k)	LazyDINO (1k)	LazyDINO (16k)
Algorithm steps	Amortized PtO evaluations (sec)	–	–	2,150/4	34,200/4
	Parallel amortized PtO evaluations (sec)	–	–	107.5/4	1,710/4
	Amortized Jacobian (sec)	–	–	220/4	3,440/4
	Parallel amortized Jacobian (sec)	–	–	11/4	172/4
	Amortized DINO training (sec)	–	–	120/4	1,840/4
	TMVI training* (sec)	2,390	38,500	750	750
	Parallel TMVI training (sec)	119.5	1,925	–	–
Total	Time per BIP (sec)	2,390	38,500	1,372.5	10,620
	Parallel time per BIP (sec)	119.5	1,925	809.625	1,680.5
	Relative mean error achieved (%)	230	90	12	3.5

Table 4: **Example II: `LazyDINO`/`LazyMap` timing comparison.** We include two sample sizes (1k and 16k) for comparison. The total sequential execution times for the same sample sizes are less for `LazyDINO` due to the amortization of the offline phase across four BIPs corresponding to four instances of observational data. The total 20-way parallel execution times are similar for `LazyMap`(16k) and `LazyDINO` (16k). For both sample sizes, the relative mean error achieved is much lower for `LazyDINO`. Moreover, `LazyDINO` (1k) achieves much smaller relative mean error than `LazyDINO` (16k) in less time. * denotes the fact that `LazyDINO` already performs batch-vectorized computation, so that parallel computation of the surrogate PtO map is not applicable.

SBAI vs. ***LazyDINO.*** In typical formulations of SBAI, sampling requires inverting the transport map.[1] For particular transport map parametrizations, such as the inverse autoregressive flows (IAFs) we used for numerical results, the scalability of this inversion can be preserved. In the case of IAFs, the cost is dominated by d_r 1-dimensional root-finding problems that can be solved relatively quickly via the bisection method. However, in contrast, sampling with `LazyDINO` involves only explicit evaluations of neural networks and produces samples in significantly less time.

In Table 5, we report average times to train and compute 1 million i.i.d. approximate posterior samples for four instances of observational data studied in Example I and II. We note that these results depend on transport map architecture and the quality of implementations; however, the overall point concerning the additional expense to invert maps in SBAI is broadly applicable. Due to these higher sampling costs, the transport map training time required by `LazyDINO` *for each* instance of observational data is amortized across the *sampling of the posterior*, such that for large enough sample sizes, it can be less time consuming to first train a `LazyDINO` and then to subsequently sample, rather than to sample using the SBAI method.

6.4. Visualization of discrepancy in marginals

In this section, we provide visual evidence of the relative performances of different methods by investigating pairwise 2D marginal kernel density estimates and 1D marginal histograms of samples produced via the posterior sampling. We use a geometric MCMC method [103, ∞-mMALA in section 3.2] to produce samples from the ground truth posterior. Marginal discrepancies provide a diagnostic measure for the quality of each posterior approximation—matching marginals is only a necessary, not sufficient, condition for matching the joint distribution. We plot a progression of marginals increasing sample size from left

[1] While this need not necessarily be the case, the alternative requires inverting the transport map *during training*, which is often considered too expensive.

Example I	**Method**	
Time (sec)	SBAI (16k)	`LazyDINO` (16k)
1 million samples	1130	60
Non-amortized training	—	460
Amortized training	930 / 4	1220/4
Total: sampling per BIP	1362.5	825

Example II	**Method**	
Time (sec)	SBAI (16k)	`LazyDINO` (16k)
1 million samples	1150	60
Non-amortized training	—	750
Amortized training	960 / 4	1840/4
Total: sampling per BIP	1390	1270

Table 5: **SBAI vs `LazyDINO` sampling times.** For large enough sample sizes, the inversion-to-sample approach of SBAI is more costly than the optimize-to-sample appraoch of `LazyDINO`. Since previous comparisons demonstrated that `LazyDINO` is consistently much more accurate on a sample cost basis, this adds to the argument for utilizing `LazyDINO` over SBAI in the setting where amortized offline computations are desirable to facilitate real-time solutions. To facilitate direct comparison, we employ identical IAF architectures for SBAI and `LazyDINO`, except that the input dimension for the conditional transport map for SBAI is $\mathbb{R}^{d_r} \times \mathbb{R}^{d_{\boldsymbol{y}}}$ rather than $\mathbb{R}^{d_r}$ in `LazyDINO`. The architecture used is the same as the one described with all LMVI methods in Section 5.4. Times are averaged across the four BIPs corresponding to four instances of observational data and rounded up to the nearest 10 seconds. We report the times for 16k training samples. However, the times are similar across all sample sizes. Since the transport map architecture was chosen to be identical, sampling times for the two examples were essentially the same. The additional time to sample with SBAI stems from the need to apply the inverse of the transport map when sampling. Amortized training refers to conditional transport map training for SBAI and `RB-DINO` training for `LazyDINO`.

to right. We investigate the different posterior marginal comparisons for Example I in Figures 12 to 15. Consistent with the previous results, `LazyDINO` produces consistently better approximations than the other TMVI methods, and overtakes the LA-baseline for a relatively small amount of training samples.

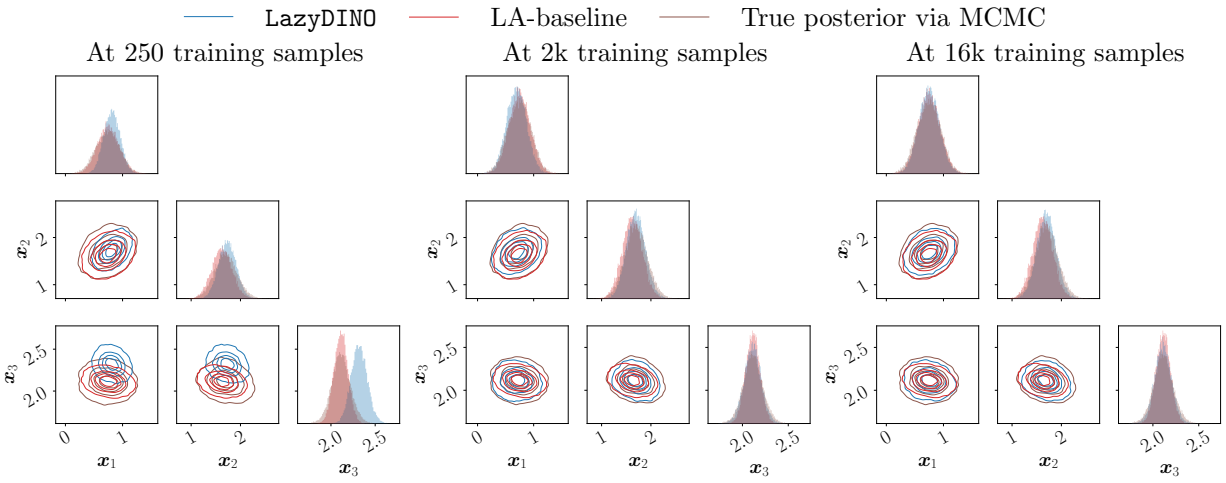

Figure 12: **Example I: `LazyDINO` vs Laplace approximation marginals.** At 250 training samples, we see clear deviation in the marginals for both approaches, though LA-baseline has contours that match the posterior marginals more closely. By 2k training samples, `LazyDINO` seems to match the marginals better, especially in the 'tail' contours of the true posterior.

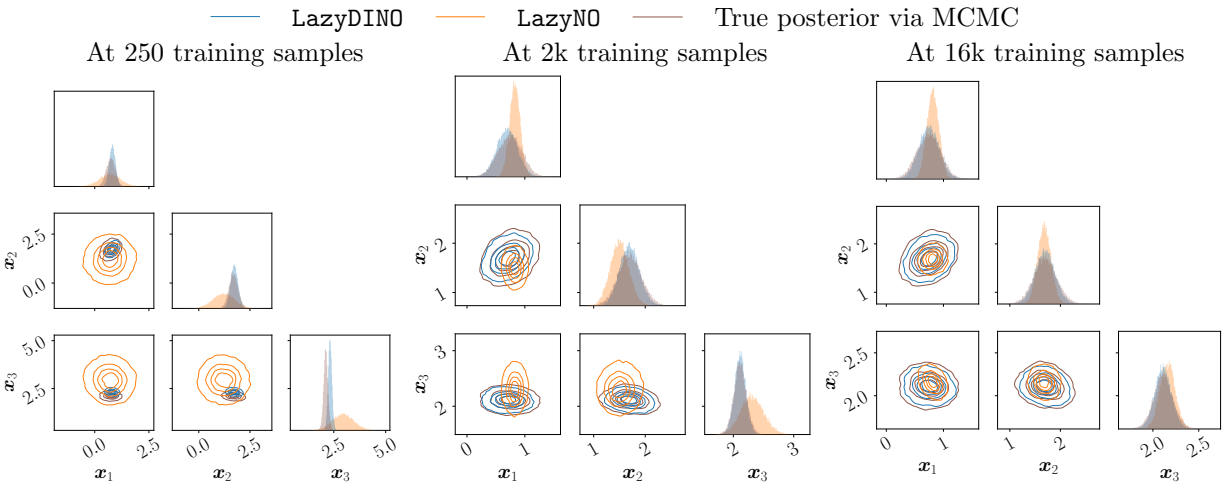

Figure 13: **Example I: `LazyDINO` vs `LazyNO` marginals.** Consistent with Figure 6 and Figure 10, we see the `LazyNO` fail to capture the posterior marginals well in all sample sizes shown. In comparison, `LazyDINO` closely matches the posterior marginals at 250 training samples (*left*)

We now investigate posterior marginals for Example II in Figures 16 to 19. For this set of results, the target marginals exhibit more non-Gaussianity than the previous example. In this set of results, `LazyDINO` consistently outperforms all other methods. Notably, due to the non-Gaussianity of the problem, the LA-baseline does not produce accurate marginal approximations, allowing the nonlinearly parametrized `LazyDINO` to overtake it for very limited sample data.

6.5. Visualization of discrepancy in mean, MAP point, and point-wise variance

In this section, we proceed with additional visualizations of the discrepancies in the mean, MAP point estimates and point-wise variances, and associated point-wise absolute errors arising from the different

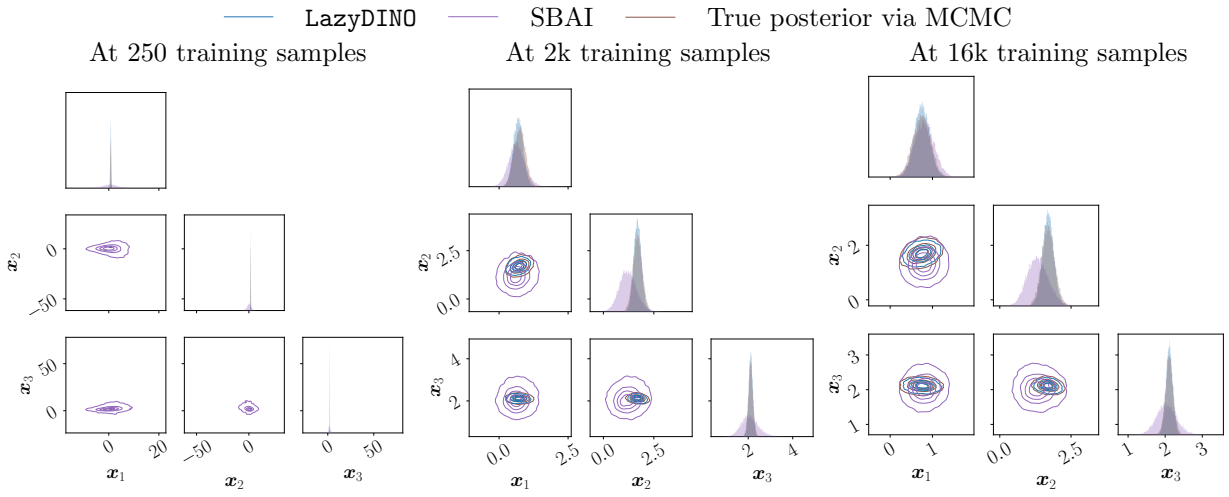

Figure 14: **Example I: `LazyDINO` vs SBAI marginals.** The marginals produced via SBAI are much further from the true posterior marginals than the other approaches. Notably, SBAI consistently overestimates the uncertainty in posterior reconstruction and still yields a poor reconstruction of posterior marginals for $16,000$ samples.

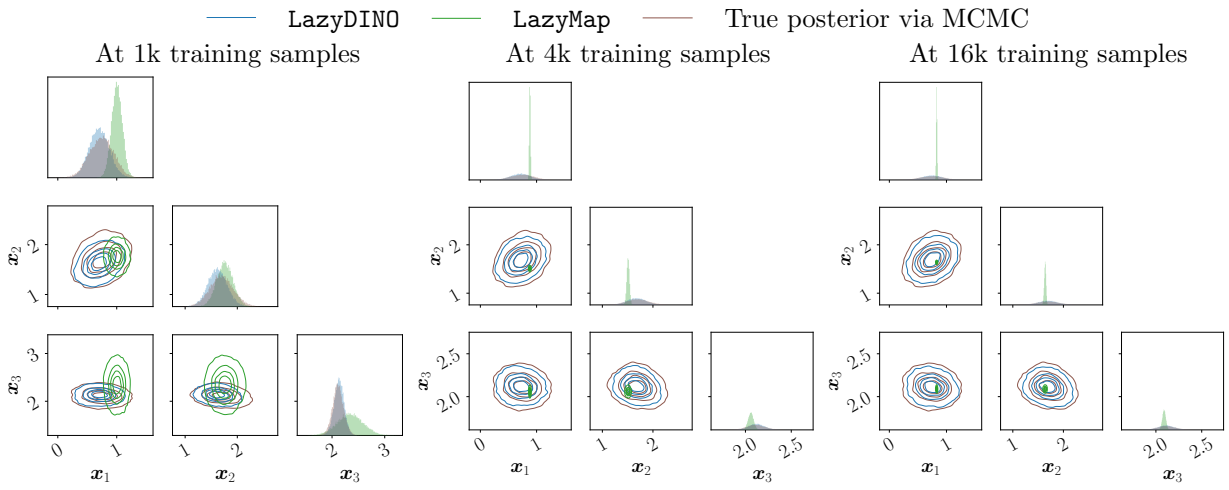

Figure 15: **Example I: `LazyDINO` vs `LazyMap` marginals.** The LazyMap marginals are quite poor for 1k training samples and only get more concentrated as the number of training samples increases. The equivalent sample-cost `LazyDINO` posterior marginals match the ground truth substantially better. Notably, `LazyMap` is highly concentrated, leading to naïve underestimation of the uncertainty in the posterior approximation. In comparison, `LazyDINO` yields a faithful approximation for only 250 training data.

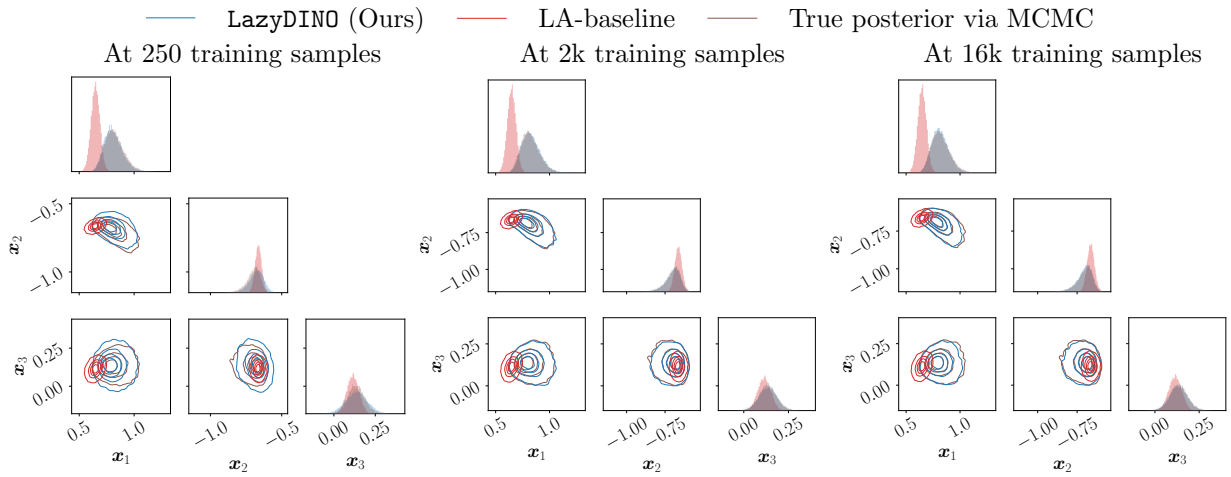

Figure 16: **Example II: `LazyDINO` vs Laplace approximation marginals.** Due to the non-Gaussianity of the posterior, Laplace approximation struggles to capture the overall behavior of the marginals depicted, whereas the `LazyDINO` marginals are close to the posterior marginals at 2k training samples. We note that a marginal of the MAP estimate is not the same as the MAP of a marginal in general, which may explain the apparent inconsistency that this marginal of the Laplace approximation is not visually centered on the region of highest probability of these particular posterior marginals.

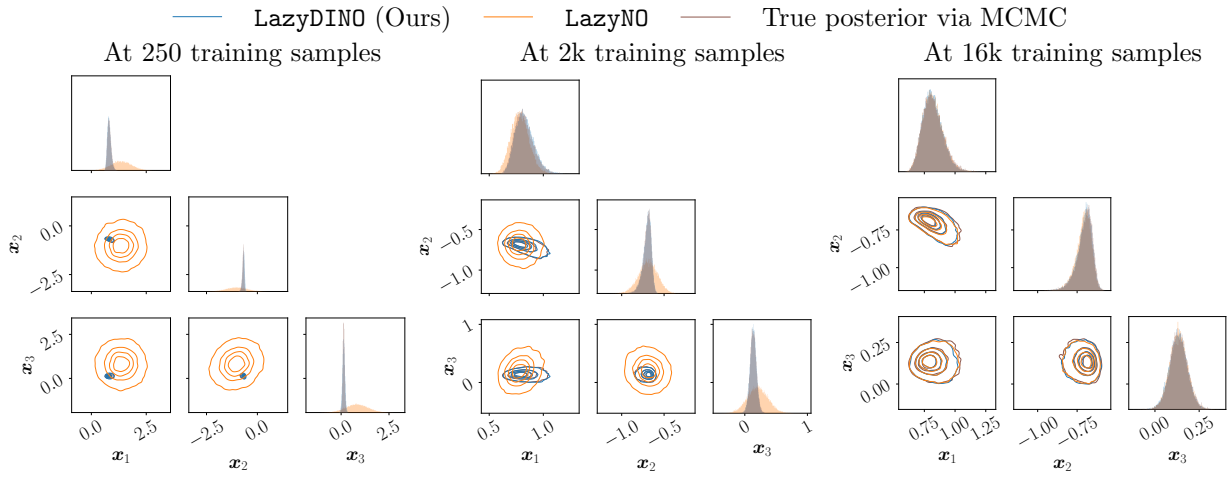

Figure 17: **Example II: `LazyDINO` vs `LazyNO` marginals.** With 250 training samples, `LazyNO` produces far underconcentrated samples in these marginals. With 16k training samples, both methods produce similar marginal distributions. The notable takeaway is that `LazyNO` requires multiple orders of magnitude more samples to have comparable performance to `LazyDINO` with 250 samples.

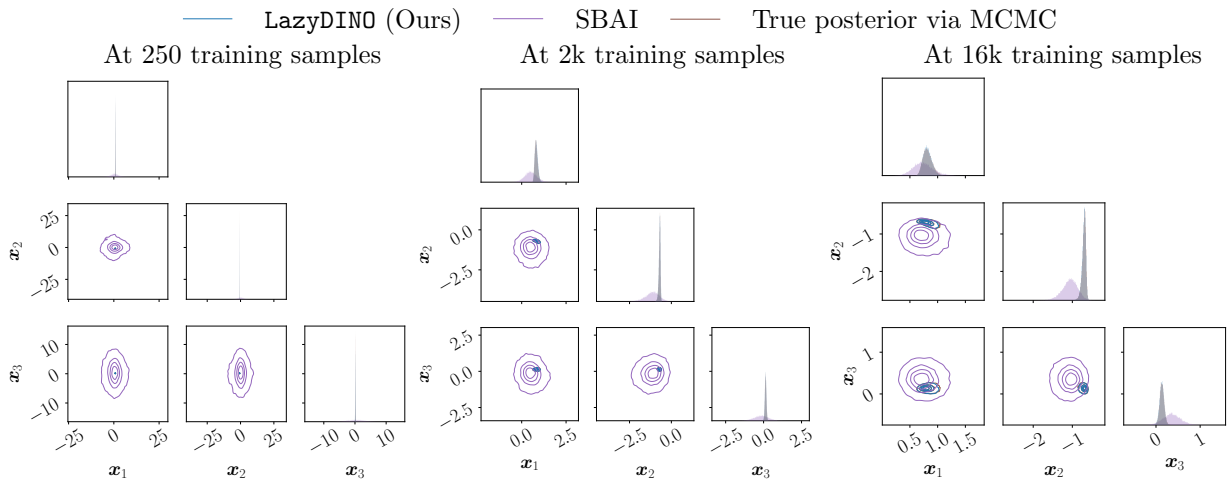

Figure 18: **Example II: `LazyDINO` vs SBAI marginals.** SBAI produces samples that are highly under-concentrated. SBAI is simultaneously off in capturing the peak locations of the marginals while overestimating the uncertainty in the parameter.

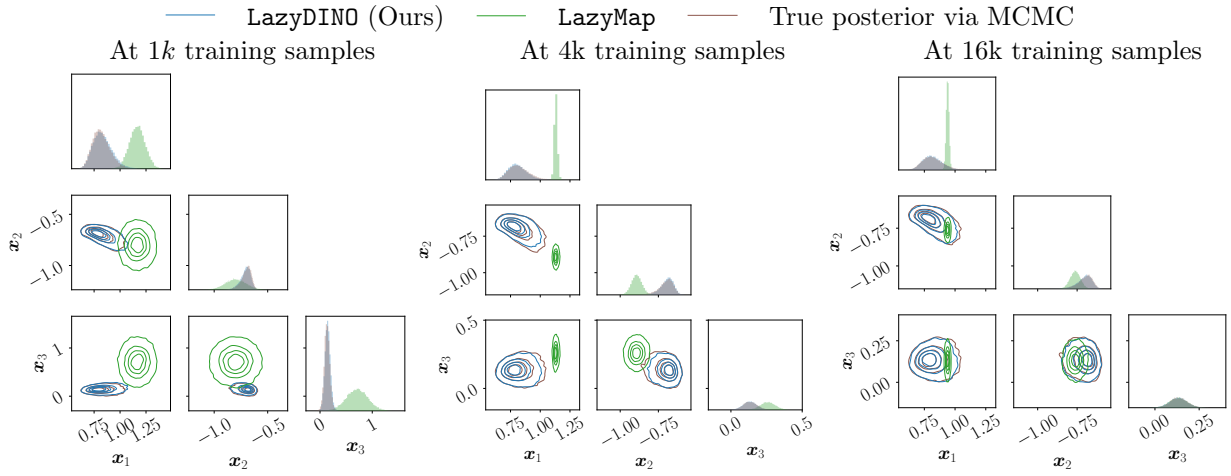

Figure 19: **Example II: `LazyDINO` vs. `LazyMap` marginals.** `LazyMap` fails to capture the contours of the posterior marginals faithfully. `LazyMap` is off by a constant error in capturing the peak of the marginals. As more data is available, it tends to over-concentrate, leading to naïve underestimation of risk.

methods for varying amounts of training data used. These comparisons allow the methods to be visually differentiated in their ability to resolve features in the parameter reconstruction via the BIP. We begin by visualizing the mean, MAP estimates, and point-wise variances for Example I in Figures 20 to 22, respectively. The general trend is consistent with the previous numerical studies: `LazyDINO` leads to superior approximation than the other methods. Notably, `LazyDINO` can capture the mean, MAP, and point-wise marginal variance somewhat faithfully for 250 samples, while other methods struggle with orders of magnitude more samples.

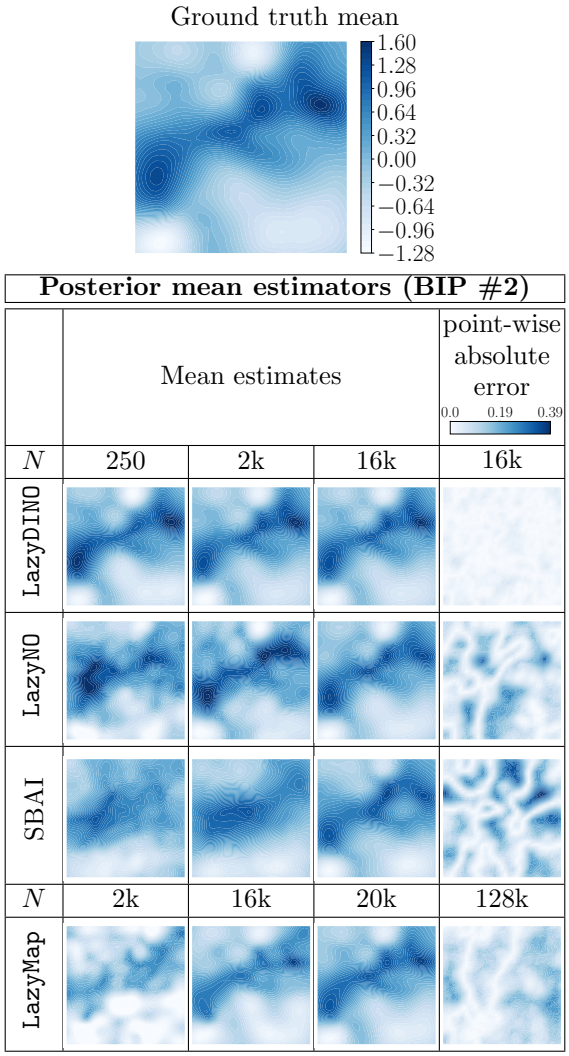

Figure 20: **Example I: progression of mean estimators.** `LazyDINO` already visually captures the mean well with 250 training samples and leads to significantly smaller point-wise errors at 16,000 samples. The next best performing method is `LazyNO`, which struggles to resolve the essential features of the mean until it has 16,000 training samples and still results in substantially higher point-wise absolute errors than `LazyDINO` at this amount of training data. The SBAI and `LazyMap` mean estimates are poor and dominated by artifacts. The mean constructions are only reasonable at the largest training sample size and still yield substantial absolute point-wise errors.

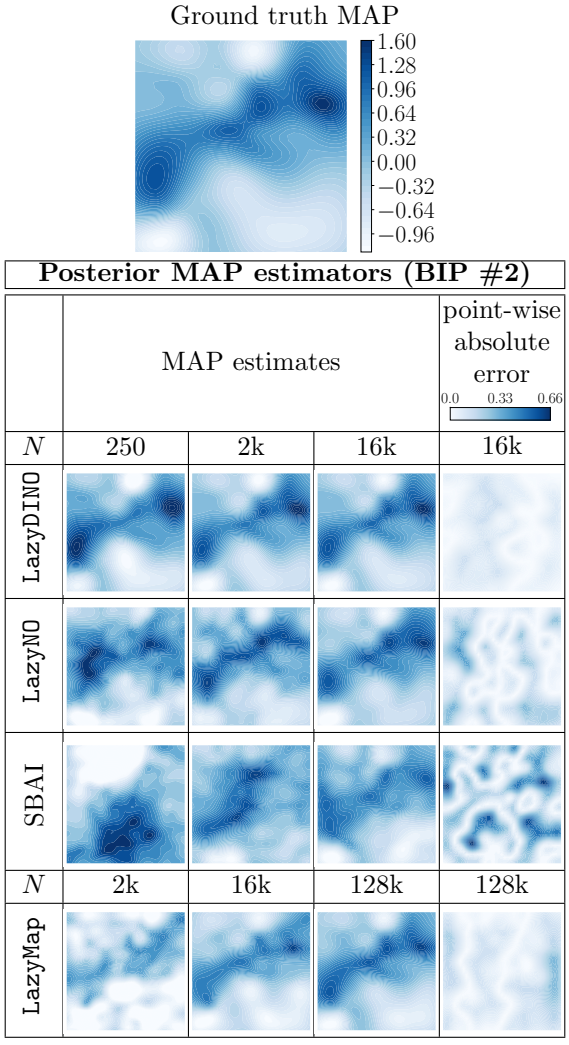

Figure 21: **Example I: progression of MAP estimators.** In a similar story to Figure 20 the `LazyDINO` MAP reconstruction is already quite accurate for 250 training data, and consistently outperforms the other methods. `LazyNO` and `LazyMap` yield reasonable MAP reconstructions given $16,000$ and $128,000$ samples respectively, albeit with higher absolute point-wise errors than `LazyDINO`. The SBAI MAP point estimate is poor even for $16,000$ training data. This is evident from the absolute point-wise errors.

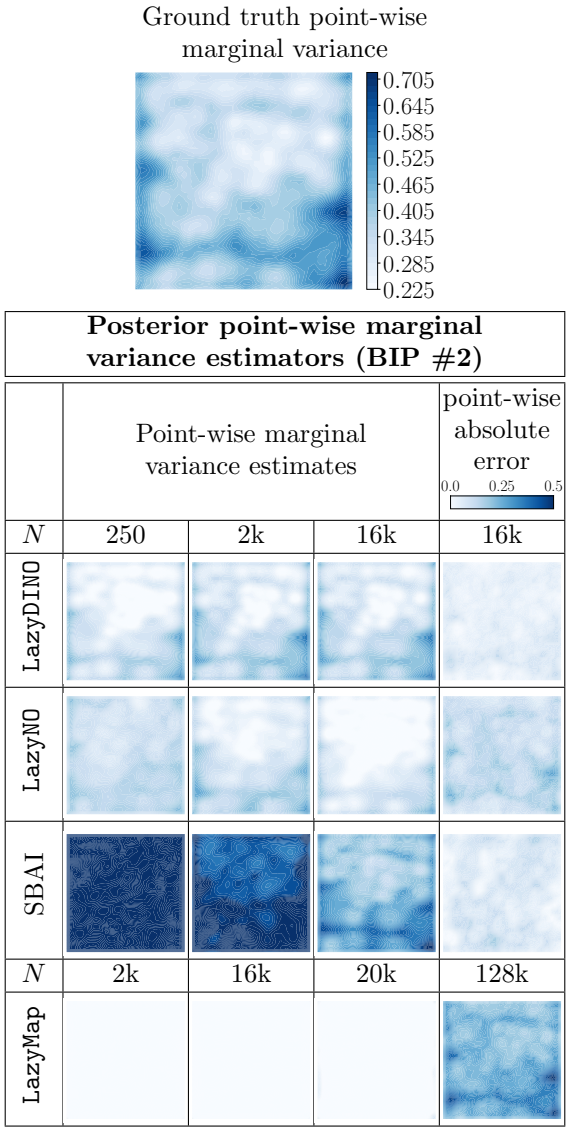

Figure 22: **Example I: progression of point-wise marginal variance estimators.** Again, the `LazyDINO` estimate of point-wise marginal variance is superior to the other methods as in the previous studies of mean and MAP estimation. Notably, SBAI overestimates the point-wise marginal variance, while `LazyMap` significantly underestimates the marginal variance. This is consistent with the evidence provided in Figures 14 and 15, which demonstrates a similar phenomenon in the marginals: SBAI is spread out while `LazyMap` is highly concentrated.

We proceed with the study of mean, MAP point, and point-wise marginal variance estimation for Example II in Figures 23 to 25. The overall story is similar to all preceding numerical studies: `LazyDINO` provides superior approximation to the other methods, and is notably able to give faithful approximations for limited samples, while the other methods require orders of magnitude more samples to achieve similar accuracy. .

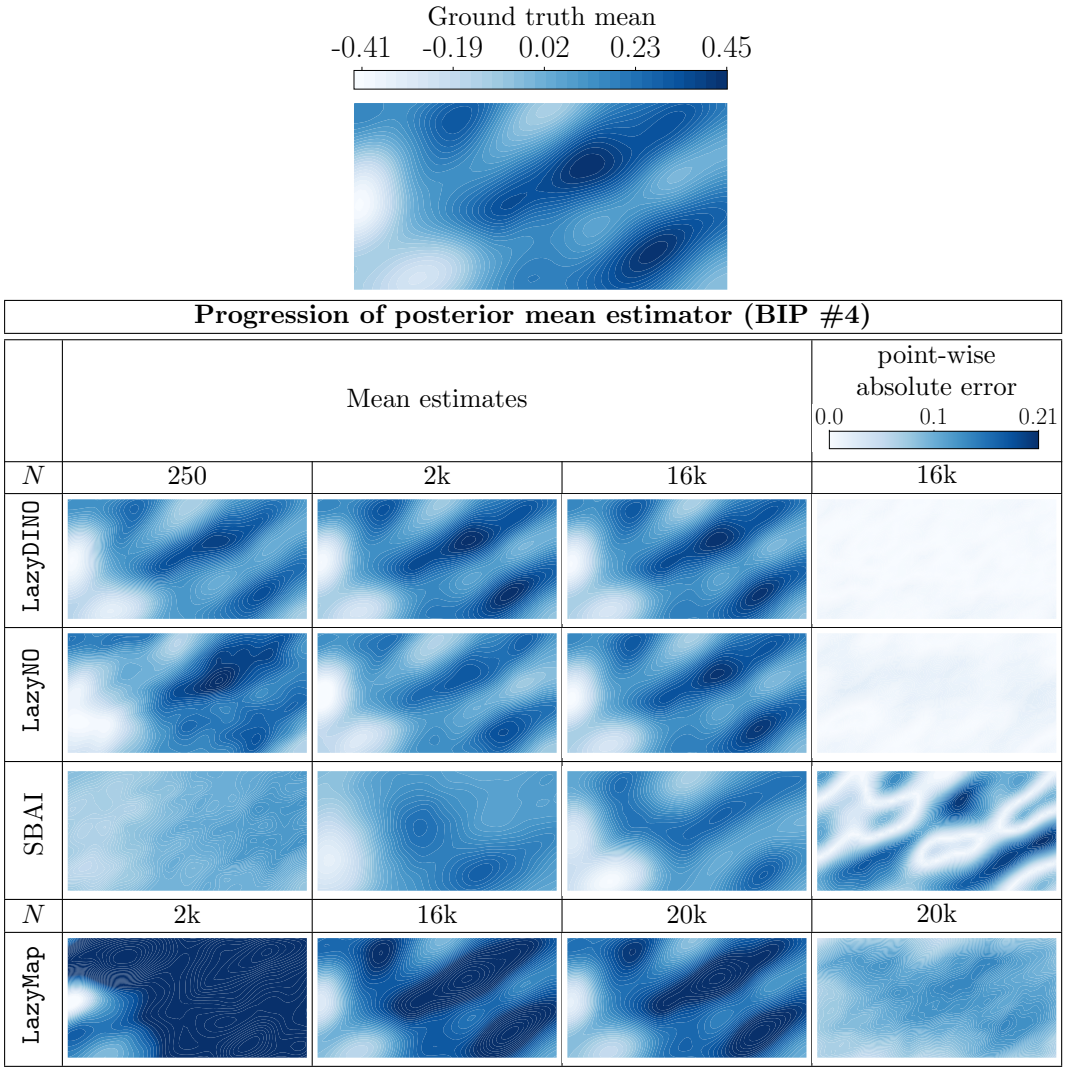

Figure 23: **Example II: progression of mean estimators with N.** As with Figure 20 we see a similar trend, where `LazyDINO` yields superior approximation than the other methods. Notably, `LazyDINO` achieves a reasonably accurate mean estimate with 250 samples. `LazyNO` eventually also yields a faithful estimate but requires an order of magnitude more samples to do so. SBAI and `LazyMAP` both struggle substantially with this problem and yield large point-wise absolute errors, even for the largest amount of training samples utilized in their construction.

This concludes our extensive numerical comparison. In almost every point of comparison `LazyDINO` yielded the most accurate estimation of the posterior distribution as evidenced by moment discrepancies, density-based diagnostics, posterior marginals, mean, MAP and point-wise marginal variance estimations. Notably, `LazyDINO` gives faithful posterior estimates for orders of magnitude fewer samples than the alternative TVMI methods. While the LA-baseline did perform well in some metrics given limited samples (e.g., MAP estimate and covariance), this approximation assumes posterior Gaussianity. It leads to constant irreducible error, making it unviable for complex nonlinear BIPs.

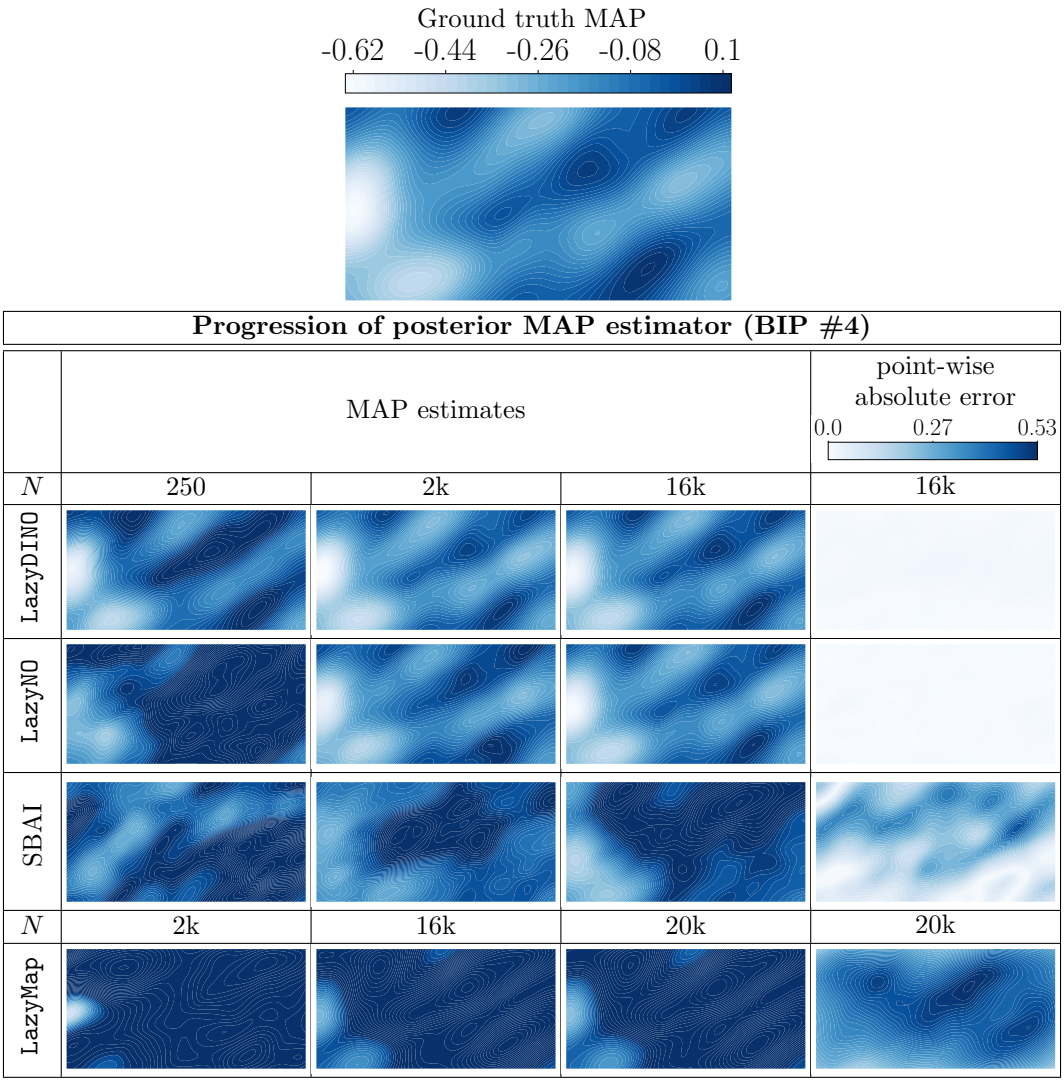

Figure 24: **Example II: progression of MAP estimators.** As with Figure 23, we see a similar story, where `LazyDINO` yields a superior approximation of the MAP point, particularly given fewer samples for its construction. Eventually, `LazyNO` yields a comparable approximation but requires an order of magnitude more samples to catch up to `LazyDINO`. Both SBAI and `LazyMap` yield poor reconstructions for the range of training samples considered in this study.

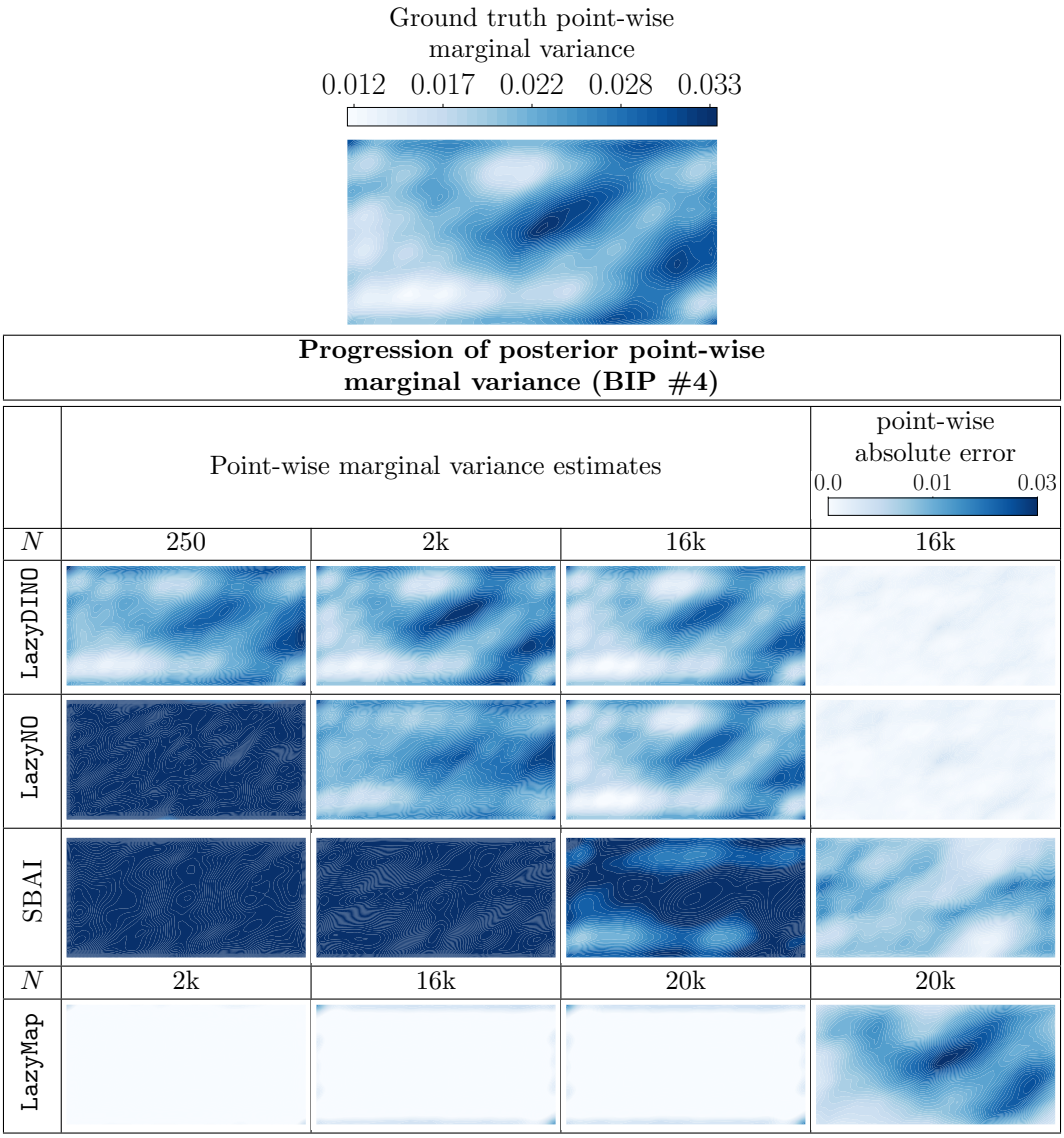

Figure 25: **Example II: progression of point-wise marginal variance estimators.** We see, yet again, the same trend of the previous studies: `LazyDINO` yields a superior approximation of the point-wise marginal variance and, notably, can deliver a faithful approximation for as little as 250 training data. `LazyNO` requires an order of magnitude more training data to catch up to the accuracy of `LazyDINO`, eventually yielding similar performance at 16,000 training data, where the approximation capabilities of this fixed architecture may have saturated. SBAI and `LazyMap`, respectively, over and underestimate point-wise variance. This observation is consistent with the spreads seen in the marginal visualizations for SBAI and `LazyMap` in Figures 18 and 19, respectively.

7. Conclusion

In this work, we present `LazyDINO`, a fast, scalable, and efficiently amortized method for high-dimensional Bayesian inversion with expensive PtO maps. The method is composed of offline and online phases. During the offline phase, we generate join samples of the PtO map and its Jacobian to construct a `RB-DINO` surrogate of the PtO map via derivative-based dimension reduction and derivative-informed learning methods. During the online phase, when observational data is given, we seek rapid posterior approximation via surrogate-driven optimization of lazy maps, i.e., structure-exploiting transport maps with relatively low-dimensional nonlinearity. The trained lazy map is used for approximate posterior sampling and density estimation.

We provide theoretical results demonstrating that the `RB-DINO` surrogate construction is optimized for amortized Bayesian inversion via lazy map variational inference. In Theorem 3.1, we show that the conventional supervised learning of the `DIPNet` surrogate architecture minimizes the upper bound on the expected error in posterior approximation when the ridge function surrogate replaces the PtO map. This architecture constricts the surrogate approximation to the parameter subspace that captures prior-to-posterior. In Theorem 3.2 and Corollary 3.3, We show that the derivative-informed learning of the surrogate minimizes the expected gradient error and optimality gap due to surrogate-driven transport map optimization. This result reflects that the surrogate Jacobian accuracy affects the quality of the trained lazy map and thus directly influences the posterior approximation accuracy.

The `LazyDINO` method has several desirable traits.

1. *Scalability.* The surrogate and transport map training in `LazyDINO` are independent of the parameter dimension as their latent representations reside in the same relatively low-dimensional derivative-informed subspace (Figures 1 and 2).

2. *Fast online inference.* Using a cheap-to-evaluate surrogate rKL objective for transport map optimization, `LazyDINO` fully exploits GPU-based accelerations to rapidly approximate posteriors (Tables 3 and 4). While our method requires solving an optimization problem to sample, we demonstrate that it leads to faster online posterior sampling than the typical inversion-to-sample approach of SBAI in the large sample size regime (Table 5).

3. *High posterior accuracy at low offline cost.* First, the RB-DINO surrogate and the lazy map are co-designed to exploit the structure of the BIP efficiently. Second, the derivative-informed learning method is highly cost-efficient and outperforms conventional supervised learning by one to two orders of magnitude (Figures 6 and 7). Consequently, `LazyDINO` requires a much smaller cost in offline computation to achieve high accuracy in online posterior approximation across multiple instances of observational.

We studied two challenging infinite-dimensional PDE-constrained BIPs, each with four different instances of observational data: (i) inferring diffusivity field in a nonlinear reaction–diffusion PDE, and (ii) inferring Young's modulus field in a hyperelastic material thin film under deformation. In both cases, we observed one to two orders of magnitude of offline cost reduction for achieving similar accuracy in posterior approximation compared to alternative amortized inference methods such as `LazyNO` and SBAI via conditional transport. Moreover, LazyDINO consistently outperforms Laplace approximation at a small offline training sample regime (250–1,000), except for covariance approximation for Example I. In contrast, LazyNO and SBAI struggle to outperform Laplace approximation and, in some cases, failed at 16,000 offline training samples.

`LazyDINO` is a powerful method for settings requiring the repeated solution of BIPs defined by the same PtO map and prior. The efficiency gains achieved via `LazyDINO` motivate further study. First, `LazyDINO` can be applied to the case of posterior approximation for multiple independent observations simultaneously. Given the difficulty associated with concentration-of-measure for posteriors for many independent observations, the potential efficiency gains could be significant. Lastly, we will explore using `LazyDINO` in real-time uncertainty quantification for risk-averse decision-making, such as optimal experimental design and optimization under uncertainty for complex physical systems.

Ackowledgement

This work was partially supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under awards DE-SC0021239 and DE-SC0023171. Thomas O'Leary-Roseberry, Joshua Chen, and Omar Ghattas are partially supported by the National Science Foundation under awards OAC-2313033 and DMS-234643, and the Air Force Office of Scientific Research under MURI grant FA9550-21-1-0084. The work of Lianghao Cao was partially supported by a Department of Defense Vannevar Bush Faculty Fellowship held by Andrew M. Stuart, and by the SciAI Center, funded by the Office of Naval Research (ONR), under Grant Number N00014-23-1-2729.

References

[1] A. M. Stuart, Inverse problems: A Bayesian perspective, Acta Numerica 19 (2010) 451–559.
[2] T. Bui-Thanh, O. Ghattas, J. Martin, G. Stadler, A computational framework for infinite-dimensional bayesian inverse problems. part i: The linearized case, with application to global seismic inversion, 2013.
[3] N. Petra, J. Martin, G. Stadler, O. Ghattas, A computational framework for infinite-dimensional bayesian inverse problems, part ii: Stochastic newton mcmc with application to ice sheet flow inverse problems, SIAM Journal on Scientific Computing 36 (2014) A1525–A1555.
[4] O. Ghattas, K. Willcox, Learning physics-based models from data: perspectives from inverse problems and model reduction, Acta Numerica 30 (2021) 445–554.
[5] M. G. Kapteyn, J. V. Pretorius, K. E. Willcox, A probabilistic graphical model foundation for enabling predictive digital twins at scale, Nature Computational Science 1 (2021) 337–347.
[6] X. Huan, J. Jagalur, Y. Marzouk, Optimal experimental design: Formulations and computations, Acta Numerica 33 (2024) 715–840.
[7] R. Baptista, Y. Marzouk, O. Zahm, On the representation and learning of monotone triangular transport maps, Foundations of Computational Mathematics (2023).
[8] Q. Liu, D. Wang, Stein variational gradient descent: A general purpose bayesian inference algorithm, 2019.
[9] P. Chen, K. Wu, J. Chen, T. O'Leary-Roseberry, O. Ghattas, Projected stein variational newton: A fast and scalable bayesian inference method in high dimensions, Advances in Neural Information Processing Systems 32 (2019).
[10] G. Detommaso, T. Cui, A. Spantini, Y. Marzouk, R. Scheichl, A Stein variational Newton method, 2018.
[11] D. J. Rezende, S. Mohamed, Variational inference with normalizing flows, arXiv:1505.05770 (2015).
[12] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, B. Lakshminarayanan, Normalizing flows for probabilistic modeling and inference, arXiv preprint arXiv:1912.02762 (2019).
[13] M. C. Brennan, D. Bigoni, O. Zahm, A. Spantini, Y. Marzouk, Greedy inference with structure-exploiting lazy maps, 2020.
[14] T. Cui, J. Martin, Y. M. Marzouk, A. Solonen, A. Spantini, Likelihood-informed dimension reduction for nonlinear inverse problems, Inverse Problems 30 (2014) 114015.
[15] P. G. Constantine, Active Subspaces, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2015.
[16] T. Bui-Thanh, O. Ghattas, Analysis of the Hessian for inverse scattering problems. Part I: Inverse shape scattering of acoustic waves, Inverse Problems 28 (2012) 055001.
[17] T. Bui-Thanh, O. Ghattas, Analysis of the Hessian for inverse scattering problems. Part II: Inverse medium scattering of acoustic waves, Inverse Problems 28 (2012) 055002.
[18] T. Bui-Thanh, O. Ghattas, Analysis of the Hessian for inverse scattering problems. Part III: Inverse medium scattering of electromagnetic waves, Inverse Problems and Imaging 7 (2013) 1139–1155.
[19] P. Chen, O. Ghattas, Hessian-based sampling for high-dimensional model reduction, International Journal for Uncertainty Quantification 9 (2019).
[20] P. Chen, U. Villa, O. Ghattas, Hessian-based adaptive sparse quadrature for infinite-dimensional bayesian inverse problems, Computer Methods in Applied Mechanics and Engineering 327 (2017) 147–172.
[21] P. H. Flath, L. C. Wilcox, V. Akçelik, J. Hill, B. van Bloemen Waanders, O. Ghattas, Fast algorithms for Bayesian uncertainty quantification in large-scale linear inverse problems based on low-rank partial Hessian approximations, SIAM Journal on Scientific Computing 33 (2011) 407–432.
[22] T. Isaac, N. Petra, G. Stadler, O. Ghattas, Scalable and efficient algorithms for the propagation of uncertainty from data through inference to prediction for large-scale problems, with application to flow of the Antarctic ice sheet, Journal of Computational Physics 296 (2015) 348–368.
[23] A. Spantini, A. Solonen, T. Cui, J. Martin, L. Tenorio, Y. Marzouk, Optimal low-rank approximations of bayesian linear inverse problems, SIAM Journal on Scientific Computing 37 (2015) A2451–A2487.
[24] T. O'Leary-Roseberry, U. Villa, P. Chen, O. Ghattas, Derivative-informed projected neural networks for high-dimensional parametric maps governed by PDEs, Computer Methods in Applied Mechanics and Engineering 388 (2022) 114199.
[25] J. Hesthaven, S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, Journal of Computational Physics 363 (2018) 55–78.
[26] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: Learning maps between function spaces with applications to PDEs, Journal of Machine Learning Research 24 (2023) 1–97.

[27] N. B. Kovachki, S. Lanthaler, A. M. Stuart, Operator learning: Algorithms and analysis, arxiv.2402.15715 (2024).
[28] D. Luo, T. O'Leary-Roseberry, P. Chen, O. Ghattas, Efficient PDE-constrained optimization under high-dimensional uncertainty using derivative-informed neural operators, arXiv preprint arXiv:2305.20053 (2023).
[29] L. Cao, T. O'Leary-Roseberry, O. Ghattas, Derivative-informed neural operator acceleration of geometric MCMC for infinite-dimensional Bayesian inverse problems, arXiv preprint arXiv:2403.08220 (2024).
[30] T. O'Leary-Roseberry, P. Chen, U. Villa, O. Ghattas, Derivative-informed neural operator: an efficient framework for high-dimensional parametric derivative learning, Journal of Computational Physics 496 (2024) 112555.
[31] T. Cui, O. Zahm, Data-free likelihood-informed dimension reduction of Bayesian inverse problems, Inverse Problems 37 (2021) 045009.
[32] O. Zahm, P. G. Constantine, C. Prieur, Y. M. Marzouk, Gradient-based dimension reduction of multivariate vector-valued functions, SIAM Journal on Scientific Computing 42 (2020) A534–A558.
[33] P. S. Laplace, Memoir on the probability of the causes of events (1774). mémoires de mathématique et de physique, tome sixième. (english translation by s. m. stigler), Statistical science 1 (1986) 364–378.
[34] T. Bui-Thanh, C. Burstedde, O. Ghattas, J. Martin, G. Stadler, L. C. Wilcox, Extreme-scale uq for bayesian inverse problems governed by pdes, in: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12, IEEE Computer Society Press, Washington, DC, USA, 2012.
[35] Y. M. Marzouk, H. N. Najm, Dimensionality reduction and polynomial chaos acceleration of bayesian inference in inverse problems, Journal of Computational Physics 228 (2009) 1862–1902.
[36] O. Zahm, T. Cui, K. Law, A. Spantini, Y. Marzouk, Certified dimension reduction in nonlinear Bayesian inverse problems, Mathematics of Computation 91 (2022) 1789–1835.
[37] D. Bigoni, Y. Marzouk, C. Prieur, O. Zahm, Nonlinear dimension reduction for surrogate modeling using gradient information, Information and Inference: A Journal of the IMA 11 (2022) 1597–1639.
[38] Y. M. Marzouk, H. N. Najm, L. A. Rahn, Stochastic spectral methods for efficient Bayesian solution of inverse problems, Journal of Computational Physics 224 (2007) 560–586.
[39] Y. Marzouk, D. Xiu, A stochastic collocation approach to bayesian inference in inverse problems, COMMUNICATIONS IN COMPUTATIONAL PHYSICS 6 (2009) 826–847.
[40] I.-G. Farcas, J. Latz, E. Ullmann, T. Neckel, H.-J. Bungartz, Multilevel adaptive sparse Leja approximations for Bayesian inverse problems, SIAM Journal on Scientific Computing 42 (2020) A424–A451.
[41] D. Galbally, K. Fidkowski, K. Willcox, O. Ghattas, Non-linear model reduction for uncertainty quantification in large-scale inverse problems, International Journal for Numerical Methods in Engineering 81 (2010) 1581–1608.
[42] C. Lieberman, K. Willcox, O. Ghattas, Parameter and state model reduction for large-scale statistical inverse problems, SIAM Journal on Scientific Computing 32 (2010) 2523–2542.
[43] T. Cui, Y. M. Marzouk, K. E. Willcox, Data-driven model reduction for the Bayesian solution of inverse problems, International Journal for Numerical Methods in Engineering 102 (2015) 966–990.
[44] B. Peherstorfer, K. Willcox, M. Gunzburger, Survey of multifidelity methods in uncertainty propagation, inference, and optimization, SIAM Review 60 (2018) 550–591.
[45] M. B. Lykkegaard, T. J. Dodwell, C. Fox, G. Mingas, R. Scheichl, Multilevel delayed acceptance MCMC, SIAM/ASA Journal on Uncertainty Quantification 11 (2023) 1–30.
[46] L. Cao, T. O'Leary-Roseberry, P. K. Jha, J. T. Oden, O. Ghattas, Residual-based error correction for neural operator accelerated infinite-dimensional Bayesian inverse problems, Journal of Computational Physics 486 (2023) 112104.
[47] K. Bhattacharya, B. Hosseini, N. B. Kovachki, A. M. Stuart, Model reduction and neural network for parametric PDEs, The SMAI Journal of computational mathematics 7 (2021) 121–157.
[48] S. Fresca, A. Manzoni, POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition, Computer Methods in Applied Mechanics and Engineering 388 (2022) 114181.
[49] T. O'Leary-Roseberry, X. Du, A. Chaudhuri, J. R. Martins, K. Willcox, O. Ghattas, Learning high-dimensional parametric maps via reduced basis adaptive residual networks, Computer Methods in Applied Mechanics and Engineering 402 (2022) 115730.
[50] L. Lu, P. Jin, G. Pang, Z. Zhang, G. E. Karniadakis, Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, Nature Machine Intelligence 3 (2021) 218–229.
[51] J. H. Seidman, G. Kissas, G. J. Pappas, P. Perdikaris, Variational autoencoding neural operators, arXiv preprint, arXiv.2302.10351 (2023).
[52] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Fourier neural operator for parametric partial differential equations, arXiv preprint, arXiv.2010.08895 (2021).
[53] Q. Cao, S. Goswami, G. E. Karniadakis, LNO: Laplace neural operator for solving differential equations, arXiv preprint, arXiv.2303.10528 (2023).
[54] S. Lanthaler, Z. Li, A. M. Stuart, The nonlocal neural operator: Universal approximation, arXiv preprint, arXiv.2304.13221 (2023).
[55] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: Graph kernel network for partial differential equations, arXiv preprint, arXiv.2003.03485 (2020).
[56] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, A. Anandkumar, Physics-informed neural operator for learning partial differential equations, ACM / IMS Journal of Data Science (2024).
[57] S. Wang, H. Wang, P. Perdikaris, Learning the solution operator of parametric partial differential equations with physics-informed DeepONets, Science Advances 7 (2021) eabi8605.
[58] J. Go, P. Chen, Accelerating Bayesian Optimal Experimental Design with Derivative-Informed Neural Operators, arXiv preprint arXiv:2312.14810 (2023).

[59] J. Go, P. Chen, Sequential infinite-dimensional Bayesian optimal experimental design with derivative-informed latent attention neural operator, arXiv preprint arXiv:2409.09141 (2024).
[60] Y. Qiu, N. Bridges, P. Chen, Derivative-enhanced deep operator network, arXiv.2402.19242 (2024).
[61] E. G. Tabak, C. V. Turner, A family of nonparametric density estimation algorithms, Communications on Pure and Applied Mathematics 66 (2013) 145–164.
[62] I. Kobyzev, S. Prince, M. Brubaker, Normalizing flows: An introduction and review of current methods, IEEE Transactions on Pattern Analysis and Machine Intelligence (2020).
[63] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real NVP, arXiv:1605.08803 (2016).
[64] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, M. Welling, Improved variational inference with inverse autoregressive flow, in: D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29, Curran Associates, Inc., 2016, pp. 4743–4751.
[65] G. Papamakarios, T. Pavlakou, I. Murray, Masked autoregressive flow for density estimation, in: Advances in Neural Information Processing Systems, pp. 2338–2347.
[66] C.-W. Huang, D. Krueger, A. Lacoste, A. Courville, Neural autoregressive flows, in: International Conference on Machine Learning, PMLR, pp. 2078–2087.
[67] N. De Cao, I. Titov, W. Aziz, Block neural autoregressive flow, arXiv preprint arXiv:1904.04676 (2019).
[68] P. Jaini, K. A. Selby, Y. Yu, Sum-of-squares polynomial flow, in: International Conference on Machine Learning, pp. 3009–3018.
[69] H. Daniels, M. Velikova, Monotone and partially monotone neural networks, IEEE Transactions on Neural Networks 21 (2010) 906–917.
[70] A. Wehenkel, G. Louppe, Unconstrained monotonic neural networks, Advances in neural information processing systems 32 (2019).
[71] H. Knothe, Contributions to the theory of convex bodies., Michigan Mathematical Journal 4 (1957) 39–52.
[72] M. Rosenblatt, Remarks on a Multivariate Transformation, The Annals of Mathematical Statistics 23 (1952) 470 – 472.
[73] T. A. El Moselhy, Y. M. Marzouk, Bayesian inference with optimal maps, Journal of Computational Physics 231 (2012) 7815–7850.
[74] Y. Marzouk, T. Moselhy, M. Parno, A. Spantini, Sampling via measure transport: An introduction, in: Handbook of Uncertainty Quantification, R. Ghanem, D. Higdon, and H. Owhadi, editors, Springer, 2016.
[75] A. Spantini, D. Bigoni, Y. Marzouk, Inference via low-dimensional couplings, The Journal of Machine Learning Research 19 (2018) 2639–2709.
[76] R. Baptista, O. Zahm, Y. Marzouk, An adaptive transport framework for joint and conditional density estimation, arXiv:2009.10303 (2020).
[77] J. Zech, Y. Marzouk, Sparse approximation of triangular transports, Part II: The infinite-dimensional case, Constr. Approx. 55 (2022) 987–1036.
[78] J. Westermann, J. Zech, Measure transport via polynomial density surrogates, arXiv preprint (2023).
[79] J. Zech, Y. Marzouk, Sparse approximation of triangular transports, Part I: The finite-dimensional case, Constr. Approx. 55 (2022) 919–986.
[80] J. Zeghal, F. Lanusse, A. Boucaud, B. Remy, E. Aubourg, Neural posterior estimation with differentiable simulators, 2022.
[81] J. Brehmer, G. Louppe, J. Pavez, K. Cranmer, Mining gold from implicit models to improve likelihood-free inference, Proceedings of the National Academy of Sciences 117 (2020) 5242–5249.
[82] C. Durkan, G. Papamakarios, I. Murray, Sequential neural methods for likelihood-free inference, arXiv preprint arXiv:1811.08723 (2018).
[83] G. Papamakarios, D. Sterratt, I. Murray, Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows, in: The 22nd International Conference on Artificial Intelligence and Statistics, PMLR, pp. 837–848.
[84] D. Greenberg, M. Nonnenmacher, J. Macke, Automatic posterior transformation for likelihood-free inference, in: International Conference on Machine Learning, PMLR, pp. 2404–2414.
[85] G. Papamakarios, I. Murray, Fast ϵ-free inference of simulation models with bayesian conditional density estimation, 2018.
[86] R. Baptista, L. Cao, J. Chen, O. Ghattas, F. Li, Y. M. Marzouk, J. T. Oden, Bayesian model calibration for block copolymer self-assembly: Likelihood-free inference and expected information gain computation via measure transport, Journal of Computational Physics 503 (2024) 112844.
[87] A. Ganguly, S. Jain, U. Watchareeruetai, Amortized variational inference: A systematic review, Journal of Artificial Intelligence Research 78 (2023) 167–215.
[88] C. Soize, R. Ghanem, Physical systems with random uncertainties: Chaos representations with arbitrary probability measure, SIAM Journal on Scientific Computing 26 (2004) 395–410.
[89] Coordinate transformation and polynomial chaos for the bayesian inference of a gaussian process with parametrized prior covariance function, Computer Methods in Applied Mechanics and Engineering 298 (2016) 205–228.
[90] D. M. Blei, A. Kucukelbir, J. D. McAuliffe, Variational inference: A review for statisticians, Journal of the American statistical Association 112 (2017) 859–877.
[91] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, M. Welling, Improving variational inference with inverse autoregressive flow, 2017.
[92] L. Bottou, F. E. Curtis, J. Nocedal, Optimization methods for large-scale machine learning, SIAM review 60 (2018) 223–311.
[93] T. Isaac, N. Petra, G. Stadler, O. Ghattas, Scalable and efficient algorithms for the propagation of uncertainty from

data through inference to prediction for large-scale problems, with application to flow of the antarctic ice sheet, Journal of Computational Physics 296 (2015) 348–368.
[94] U. Villa, N. Petra, O. Ghattas, hIPPYlib: An extensible software framework for large-scale inverse problems governed by PDEs: Part I: Deterministic inversion and linearized Bayesian inference, ACM Transactions on Mathematical Software 47 (2021).
[95] U. Villa, T. O'Leary-Roseberry, A note on the relationship between pde-based precision operators and mat\'ern covariances, arXiv preprint arXiv:2407.00471 (2024).
[96] J. Kirchhoff, D. Luo, T. O'Leary-Roseberry, O. Ghattas, Inference of Heterogeneous Material Properties via Infinite-Dimensional Integrated DIC, arXiv preprint arXiv:2408.10217 (2024).
[97] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017.
[98] A. Beskos, G. Roberts, A. Stuart, J. Voss, MCMC methods for diffusion bridges, Stochastics and Dynamics 8 (2008) 319–350.
[99] S. Agapiou, O. Papaspiliopoulos, D. Sanz-Alonso, A. M. Stuart, Importance sampling: Intrinsic dimension and computational cost, 2017.
[100] B. Li, T. Bengtsson, P. Bickel, Curse-of-dimensionality revisited: Collapse of importance sampling in very large scale systems (2005).
[101] F. M. Polo, R. Vicente, Effective sample size, dimensionality, and generalization in covariate shift adaptation, Neural Computing and Applications 35 (2020) 18187–18199.
[102] D. Sanz-Alonso, Z. Wang, Bayesian update with importance sampling: Required sample size, Entropy 23 (2020) 22.
[103] A. Beskos, M. Girolami, S. Lan, P. E. Farrell, A. M. Stuart, Geometric MCMC for infinite-dimensional inverse problems, Journal of Computational Physics 335 (2017) 327–351.
[104] D. Nualart, The Malliavin calculus and related topics, volume 1995, Springer, 2006.
[105] V. I. Bogachev, Gaussian measures, 62, American Mathematical Soc., 1998.
[106] N. Halko, P.-G. Martinsson, J. A. Tropp, Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions, 2010.
[107] H. Xiang, J. Zou, Randomized algorithms for large-scale inverse problems with general regularizations, 2014.
[108] A. K. Saibaba, J. Lee, P. K. Kitanidis, Randomized algorithms for generalized hermitian eigenvalue problems with application to computing karhunen-loève expansion, 2015.
[109] G. H. Golub, Q. Ye, An inverse free preconditioned krylov subspace method for symmetric generalized eigenvalue problems, SIAM Journal on Scientific Computing 24 (2002) 312–334.
[110] Y. Saad, Krylov subspace methods for solving large unsymmetric linear systems, Mathematics of Computation 37 (1981) 105–126.
[111] D. C. Sorensen, Truncated qz methods for large scale generalized eigenvalue problems, Electron. Trans. Numer. Anal. 7 (1998) 141–162.
[112] J. van den Eshof, G. L. G. Sleijpen, Inexact krylov subspace methods for linear systems, SIAM Journal on Matrix Analysis and Applications 26 (2004) 125–153.
[113] P. Chowdhury, The truncated lanczos algorithm for partial solution of the symmetric eigenproblem, Computers & Structures 6 (1976) 439–446.
[114] Z. Yao, A. Gholami, S. Shen, M. Mustafa, K. Keutzer, M. Mahoney, Adahessian: An adaptive second order optimizer for machine learning, in: proceedings of the AAAI conference on artificial intelligence, volume 35, pp. 10665–10673.
[115] T. O'Leary-Roseberry, R. Bollapragada, Fast Unconstrained Optimization via Hessian Averaging and Adaptive Gradient Sampling Methods, arXiv preprint arXiv:2408.07268 (2024).
[116] R. Bollapragada, R. Byrd, J. Nocedal, Adaptive sampling strategies for stochastic optimization, SIAM Journal on Optimization 28 (2018) 3312–3343.
[117] D. Newton, R. Bollapragada, R. Pasupathy, N. K. Yip, A retrospective approximation approach for smooth stochastic optimization, 2024.
[118] R. Kretschmann, Are Minimizers of the Onsager–Machlup Functional Strong Posterior Modes?, SIAM/ASA Journal on Uncertainty Quantification 11 (2023) 1105–1138.
[119] R. S. Dembo, S. C. Eisenstat, T. Steihaug, Inexact newton methods, SIAM Journal on Numerical Analysis 19 (1982) 400–408.
[120] S. C. Eisenstat, H. F. Walker, Choosing the forcing terms in an inexact newton method, SIAM Journal on Scientific Computing 17 (1996) 16–32.

Appendix A. Glossary of terminology

amortized cost: Discounted cost, by spreading it out over the solution of additional problems. The more problems solved, the cheaper the amortized cost.

ANIS: Auto-normalized importance sampling, also known as self-normalized importance sampling, [99].

SBAI: Simulation-based amortized inference [87].

BIP: Bayesian inverse problem, also known as Bayesian inference problem. *Inverse* problems are typically governed by physics.

ESS: Effective sample size, an estimation of the number of independent samples drawn from a proposal distribution to obtain an expectation estimator with equivalence variance as standard Monte Carlo.

fKL: forward Kullback-Leibler. The fKL divergence is the Kullback-Leibler divergence of the approximate distribution μ from the target distribution ν, $\mathcal{D}_{\mathrm{KL}}(\nu||\mu)$.

`LazyDINO/NO`: The name of our algorithm, referring to a hybrid of `LazyMap` and the (DI)NO, (derivative-informed) neural operator trained surrogate PtO map.

`LazyMap`: A transport map that acts only in a subspace of the input, by way of a linear projection-based dimension reduction [13]. It is also used to refer to the algorithm to train it.

LMVI: Lazy map variational inference. Variational inference using structure-exploiting transport map with relatively low-dimensional nonlinearity [13].

MAP: Maximum A-Posteriori, a/the MAP estimate is a/the point of highest probability concentration of a distribution.

MC: Monte Carlo, which will always refer specifically to an i.i.d. sampling based approach to approximating an expectation.

MCMC: Markov chain Monte Carlo.

PtO: parameter-to-observable. A PtO map is a function taking the parameter we wish to infer to the non-noisy *observable*. In contrast, *observations* are measurements of the observable corrupted via noise.

rKL: reverse Kullback-Leibler. The rKL divergence is the Kullback-Leibler divergence of the target distribution ν from an approximate distribution μ, $\mathcal{D}_{\mathrm{KL}}(\mu||\nu)$.

TMVI: transport map variational inference, as in [74]. Though most modern variational inference is transport map variational inference, we use this term to *distinguish* from other forms of variational inference, i.e., inference within families of distributions that do not originate via the transport of a reference distribution.

Appendix B. The single-sample estimate of the gradient of the optimal ridge function

Proof. Given $\mu = \mu_r \otimes \mu_\perp$, where $\mu_r = \mathcal{P}_\sharp \mu$ and $\mu_\perp = (\mathrm{Id}_{\mathscr{M}} - \mathcal{P})_\sharp \mu$, we reminder the reader that the projector is defined in terms of the $H_{\mathcal{C}}$-orthonormal reduced basis $\Psi_r = \{\psi_j\}_{j=1}^{d_r}$, with $\mathrm{span}(\Psi_r) = \mathrm{Im}(\mathcal{P})$. Using the compact notation $\mathbb{E}[\boldsymbol{\mathcal{G}}|\sigma(\mathcal{P})](m_r) := \mathbb{E}_{m_\perp \sim \mu_\perp}[\boldsymbol{\mathcal{G}}(m_r + m_\perp)]$ for $m_r \in \mathrm{Im}(\mathcal{P})$ for the conditional expectation, and noting that $\mathcal{D}_r \boldsymbol{z} \overset{d}{=} m_r \overset{d}{=} \mathcal{P}m$, and $\boldsymbol{V}^* \widetilde{\boldsymbol{\mathcal{G}}}_{\mathrm{opt}}(\mathcal{D}_r \boldsymbol{z}) = \mathbf{g}_{\mathrm{opt}}(\boldsymbol{z}, \boldsymbol{w})$ we have that

$$\mathbb{E}_{m_r \sim \mu_r}\Big[\Big\| \mathbb{E}[\boldsymbol{\mathcal{G}}|\sigma(P)](m_r) - \widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}(m_r) \Big\|^2_{\Gamma_n^{-1}} + \Big\| D_H\left(\mathbb{E}[\boldsymbol{\mathcal{G}}|\sigma(P)](m_r)\right) - D_H \widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}(m_r) \Big\|^2_{\mathrm{HS}(H_{\mathcal{C}}, H_{\Gamma_n})} \Big] \tag{B.1}$$

$$= \mathbb{E}_{m \sim \mu}\Big[\Big\| \boldsymbol{\mathcal{G}}_{\mathrm{opt}}(\mathcal{P}m) - \widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}(\mathcal{P}m) \Big\|^2_{\Gamma_n^{-1}} + \Big\| D_H \boldsymbol{\mathcal{G}}_{\mathrm{opt}}(\mathcal{P}m) - D_H(\widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}} \circ \mathcal{P})(m) \Big\|^2_{\mathrm{HS}(H_{\mathcal{C}}, H_{\Gamma_n})} \Big] \tag{B.2}$$

$$\overset{d}{=} \mathbb{E}_{\boldsymbol{z} \sim \pi}\Big[\big\| \mathbf{g}_{\mathrm{opt}}(\boldsymbol{z}) - \mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z}) \big\|^2 + \Big\| \mathbb{E}[D_H \boldsymbol{\mathcal{G}}|\sigma(P)](\mathcal{D}_r \boldsymbol{z}) - D_H \widetilde{\boldsymbol{\mathcal{G}}}_{\boldsymbol{w}}(\mathcal{D}_r \boldsymbol{z}) \Big\|^2_{\mathrm{HS}(H_{\mathcal{C}}, H_{\Gamma_n})} \Big] \tag{B.3}$$

For the last line, since $\boldsymbol{\mathcal{G}} \in H^1_\mu(\mathscr{M}; H_{\Gamma_n})$, the conditional expectation also belongs to the space, $\mathbb{E}[\boldsymbol{\mathcal{G}}|\sigma(\mathcal{P})] \in H^1_\mu(\mathscr{M}; H_{\Gamma_n})$, by an isometry property (Proposition 1.2.8, [104]), and the usual commuting property between conditional expectation and differentiation apply—the Malliavin derivative of the conditional expectation with respect to the sigma algebra generated by the $\mathcal{P}$ is the conditional expectation of the Malliavin derivative. Then, $D_H \boldsymbol{\mathcal{G}}_{\mathrm{opt}}(\mathcal{P}m) := D_H \mathbb{E}[\boldsymbol{\mathcal{G}}|\sigma(\mathcal{P})](\mathcal{P}m) = \mathbb{E}[D_H \boldsymbol{\mathcal{G}}|\sigma(\mathcal{P})](\mathcal{P}m)$ a.e. Lastly, we use a change-of-variables.

In particular, we have the connection to the π-weighted Sobolev norm $H^1_\pi(\mathbb{R}^{d_r};\mathbb{R}^{d_y})$ objective:

$$\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\mathbf{g}_{\text{opt}}(\boldsymbol{z})-\mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|^2+\left\|\nabla\mathbf{g}_{\text{opt}}(\boldsymbol{z})-\nabla_{\boldsymbol{z}}\mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|_F^2\right] \tag{B.4}$$

$$=\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\boldsymbol{V}^*\mathbb{E}_{m_\perp\sim\mu_\perp}\left[\boldsymbol{\mathcal{G}}(\mathcal{D}_r\boldsymbol{z}+m_\perp)\right]-\mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|^2+\left\|\boldsymbol{V}^*\mathbb{E}_{m_\perp\sim\mu_\perp}\left[D_H\boldsymbol{\mathcal{G}}(\mathcal{D}_r\boldsymbol{z}+m_\perp)\right]\circ\mathcal{D}_r-\nabla_{\boldsymbol{z}}\mathbf{g}_{\boldsymbol{w}}(\boldsymbol{z})\right\|_F^2\right] \tag{B.5}$$

where the second term follows from a chain rule, which produces $\mathcal{D}_r$. Now, approximating this objective with a single sample $m_\perp\sim\mu_\perp$ for each $z\sim\pi$ produces our desired single-sample estimate result. □

Appendix C. Transport map variational inference on Hilbert spaces [105, section 6.6]

Let us consider the rKL objective using nonlinear transformation of Gaussian measure $\mu=\mathcal{N}(0,\mathcal{C})$ on a separable Hilbert space $\mathscr{M}$. Let $\mathcal{T}=\mathrm{Id}_{\mathscr{M}}+\mathcal{K}$ be the transport map where $\mathcal{K}:\mathscr{M}\to H_{\mathcal{C}}$ is nonlinear operator with a stochastic derivative $D_H\mathcal{K}:\mathscr{M}\to\mathrm{HS}(H_{\mathcal{C}})$ such that $D_H\mathcal{T}\in\mathrm{HS}(H_{\mathcal{C}})$ is invertible μ-a.e. Then we have

$$\mathcal{D}_{\mathrm{KL}}(\mu||\mathcal{T}^{\#}\mu^{\boldsymbol{y}})=\int_{\mathscr{M}}\log\left(\frac{\mathrm{d}\mu}{\mathrm{d}\left(\mu^{\boldsymbol{y}}\circ\mathcal{T}\right)}(m)\right)\mathrm{d}\mu(m)\,.$$

Here, we derive the density between the pullback measure and the prior. Let $\mathscr{A}$ be any measurable subset of $\mathscr{M}$, we have

$$(\mu^{\boldsymbol{y}}\circ\mathcal{T})(\mathscr{A})=\int_{\mathcal{T}(\mathscr{A})}\mathrm{d}\mu^{\boldsymbol{y}}(m)=\frac{1}{Z^{\boldsymbol{y}}}\int_{\mathscr{A}}\exp\left(-(\Phi^{\boldsymbol{y}}\circ\mathcal{T})(m)\right)\mathrm{d}(\mu\circ\mathcal{T})(m)$$

The existence and the formula of the Radon–Nikodym derivative between $\mu\circ\mathcal{T}$ and μ is non-trivial; see [105, section 6.6] for details. In the case where $D_H\mathcal{K}(m)$ is a trace class operator on $H_{\mathcal{C}}$ μ-a.e., we have

$$\frac{\mathrm{d}(\mu\circ\mathcal{T})}{\mathrm{d}\mu}(m)=\det{}_{H_{\mathcal{C}}}(D_H\mathcal{T})\exp(-\left\langle\mathcal{K}(m),m\right\rangle_{\mathcal{C}^{-1}}-\frac{1}{2}\left\|\mathcal{K}(m)\right\|^2_{\mathcal{C}^{-1}}),$$

where the determinant is taken as the product of eigenvalues with $H_{\mathcal{C}}$-orthonormal eigenbases.

Therefore, the transport objective is given by

$$\mathcal{D}_{\mathrm{KL}}(\mu||\mathcal{T}^{\#}\mu^{\boldsymbol{y}})=\mathbb{E}_{m\sim\mu}\left[(\Phi^{\boldsymbol{y}}\circ\mathcal{T})(m)-\log\det{}_{H_{\mathcal{C}}}(D_H\mathcal{T})+\left\langle\mathcal{K}m,m\right\rangle_{\mathcal{C}^{-1}}+\frac{1}{2}\left\|\mathcal{K}(m)\right\|^2_{\mathcal{C}^{-1}}\right]+C_1$$

Let us consider the perturbation of the identity $\mathcal{K}$ only acting on the latent space $\mathrm{Im}(\mathcal{E}_r)=\mathbb{R}^{d_r}$ of the projection $\mathcal{P}=\mathcal{D}_r\circ\mathcal{E}_r$ with $\mathcal{E}_r\circ\mathcal{D}_r=\mathrm{Id}_{\mathbb{R}^r}$. Then we have the following alternative definition of lazy map through $\mathcal{K}$:

$$\mathcal{K}=\underbrace{\mathcal{D}_r\circ(\mathbf{T}-\mathrm{Id}_{\mathbb{R}^{d_r}})\circ\mathcal{E}_r}_{\substack{\text{Perturbation of the identity}\\ \text{transport in }\mathrm{Im}(\mathcal{P})}},\quad \mathcal{T}=\mathrm{Id}_{\mathscr{M}}+\mathcal{K},$$

where $\mathcal{D}_r$ and $\mathcal{E}_r$ consists of $H_{\mathcal{C}}$-orthonormal reduced bases. In this case, we have

$$\begin{aligned}
\log\det(D_H\mathcal{T}(m))&\implies\log\det(\nabla\mathbf{T}(\mathcal{E}_rm)),\\
\left\langle\mathcal{K}(m),m\right\rangle_{\mathcal{C}^{-1}}&\implies(\mathcal{E}_rm)^\top\mathbf{T}(\mathcal{E}_rm)-\left\|\mathcal{E}_rm\right\|^2,\\
\frac{1}{2}\left\|\mathcal{K}(m)\right\|^2_{\mathcal{C}^{-1}}&\implies\frac{1}{2}\left\|\mathbf{T}(\mathcal{E}_rm)\right\|^2+\frac{1}{2}\left\|\mathcal{E}_rm\right\|^2-(\mathcal{E}_rm)^\top\mathbf{T}(\mathcal{E}_rm).
\end{aligned}$$

Therefore we have

$$\begin{aligned}
\mathcal{D}_{\mathrm{KL}}(\mu||\mathcal{T}^{\#}\mu^{\boldsymbol{y}})&=\mathbb{E}_{m\sim\mu}\left[(\Phi^{\boldsymbol{y}}\circ\mathcal{T})(m)-\log\det(\nabla\mathbf{T}(\mathcal{E}_rm))+\frac{1}{2}\left\|\mathbf{T}(\mathcal{E}_rm)\right\|^2\right]+C_2\\
&=\mathbb{E}_{(\boldsymbol{z},m_\perp)\sim\pi\otimes\mu_\perp}\left[(\Phi^{\boldsymbol{y}}\left((\mathcal{D}_r\circ\mathbf{T})\left(\boldsymbol{z}\right)+m_\perp\right)-\log\det(\nabla\mathbf{T}(\boldsymbol{z}))+\frac{1}{2}\left\|\mathbf{T}(\boldsymbol{z})\right\|^2\right]+C_2,
\end{aligned} \tag{C.1}$$

where $\mu_\perp=(\mathcal{I}-\mathcal{P})_{\#}\mu$ is the pushforward measure in the complimentary space.

Appendix D. Proofs of Theorems 3.1 and 3.2 and Corollary 3.3

Appendix D.1. Proof of Theorem 3.1

Proof. Proposition 4.1 by Cui and Zahm [31] gives us the following equality:

$$\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\mathcal{D}_{\mathrm{KL}}(\mu^{\boldsymbol{y}}||\widetilde{\mu}^{\boldsymbol{y}})\right] = \frac{1}{2}\mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{\mathcal{G}}(m) - \widetilde{\boldsymbol{\mathcal{G}}}(\mathcal{P}m)\right\|^2_{\Gamma_n^{-1}}\right] - \mathcal{D}_{\mathrm{KL}}(\gamma||\widetilde{\gamma}).$$

Using triangle and Cauchy–Schwarz inequalities, we have

$$\frac{1}{2}\mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{\mathcal{G}}(m) - \widetilde{\boldsymbol{\mathcal{G}}}(\mathcal{P}m)\right\|^2_{\Gamma_n^{-1}}\right] \leq \mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{\mathcal{G}}(m) - \widetilde{\boldsymbol{\mathcal{G}}}_{\mathrm{opt}}(\mathcal{P}m)\right\|^2_{\Gamma_n^{-1}}\right] + \mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\mathbf{g}_{\mathrm{opt}}(\boldsymbol{z}) - \mathbf{g}(\boldsymbol{z})\right\|^2\right].$$

Furthermore, Proposition 7 in Cao et al. [29] gives the following upper bound:

$$\mathbb{E}_{m\sim\mu}\left[\left\|\boldsymbol{\mathcal{G}}(m) - \widetilde{\boldsymbol{\mathcal{G}}}_{\mathrm{opt}}(\mathcal{P}m)\right\|^2_{\Gamma_n^{-1}}\right] \leq \mathrm{Tr}_{H_{\mathcal{C}}}\left(\left(\mathrm{Id}_{H_{\mathcal{C}}} - \mathcal{P}\right)\mathcal{H}_A\left(\mathrm{Id}_{H_{\mathcal{C}}} - \mathcal{P}\right)\right),$$

which completes the proof. □

Appendix D.2. Proof

Proof of Theorem 3.2. First, since the density between $\mathbf{T}_{\boldsymbol{\theta}\sharp}\pi$ and π is essentially bounded we have the following bound for any $f \in L^1(\pi)$ due to a change-of-variables formula and the Hölder inequality:

$$\mathbb{E}_{\boldsymbol{x}\sim\mathbf{T}_{\boldsymbol{\theta}\sharp}\pi}\left[|f(\boldsymbol{x})|\right] \leq C_1\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[|f(\boldsymbol{z})|\right],$$

where C_1 is the essential supremum of the density. Next, we have

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}}\mathcal{L}_1^{\boldsymbol{y}}(\boldsymbol{z},\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}\widetilde{\mathcal{L}}_1^{\boldsymbol{y}}(\boldsymbol{z},\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})^\top(\nabla\mathbf{g}_{\mathrm{opt}}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z})^\top\left((\mathbf{g}_{\mathrm{opt}}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - \boldsymbol{V}^*\boldsymbol{y}\right)\\
&\quad - \nabla_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})^\top(\nabla\mathbf{g}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z})^\top\left((\mathbf{g}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - \boldsymbol{V}^*\boldsymbol{y}\right)\\
&= \nabla_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})^\top\left((\nabla\mathbf{g}_{\mathrm{opt}}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - (\nabla\mathbf{g}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z})\right)^\top\left((\mathbf{g}_{\mathrm{opt}}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - \boldsymbol{V}^*\boldsymbol{y}\right)\\
&\quad + \nabla_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})^\top(\nabla\mathbf{g}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z})^\top\left((\mathbf{g}_{\mathrm{opt}}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - (\mathbf{g}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z})\right).
\end{aligned}$$

By Jensen's, triangle, and Hölder's inequalities, we have

$$\begin{aligned}
\left(\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\left\|\nabla_{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}\widetilde{\mathcal{L}}^{\boldsymbol{y}}(\boldsymbol{\theta})\right\|\right]\right)^2 &\leq \left(\mathbb{E}_{(\boldsymbol{y},\boldsymbol{z})\sim\gamma\otimes\pi}\left[\left\|\nabla_{\boldsymbol{\theta}}\mathcal{L}_1^{\boldsymbol{y}}(\boldsymbol{z},\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}\widetilde{\mathcal{L}}_1^{\boldsymbol{y}}(\boldsymbol{z},\boldsymbol{\theta})\right\|\right]\right)^2\\
&\leq \max\{C_2,C_3\}\mathbb{E}_{\boldsymbol{x}\sim\mathbf{T}_{\boldsymbol{\theta}\sharp}\pi}\left[\left\|\mathbf{g}_{\mathrm{opt}}(\boldsymbol{x}) - \mathbf{g}(\boldsymbol{x})\right\|^2 + \left\|\nabla\mathbf{g}_{\mathrm{opt}}(\boldsymbol{x}) - \nabla\mathbf{g}(\boldsymbol{x})\right\|_F^2\right],
\end{aligned}$$

where the constants are given by

$$\begin{aligned}
C_2 &= \mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\|(\nabla\mathbf{g}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z})\partial_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})\|^2\right],\\
C_3 &= \mathbb{E}_{(\boldsymbol{y},\boldsymbol{z})\sim\gamma\otimes\pi}\left[\left\|\nabla_{\boldsymbol{\theta}}\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{z})^\top\left((\mathbf{g}_{\mathrm{opt}}\circ\mathbf{T}_{\boldsymbol{\theta}})(\boldsymbol{z}) - \boldsymbol{V}^*\boldsymbol{y}\right)\right\|^2\right].
\end{aligned}$$

Since $\boldsymbol{\mathcal{G}}, \widetilde{\boldsymbol{\mathcal{G}}}\circ\mathcal{P} \in H^1_\mu(\mathscr{M}; H_{\Gamma_n})$, we have $\mathbf{g}_{\mathrm{opt}}, \mathbf{g} \in H^1_\pi(\mathbb{R}^{d_r}; \mathbb{R}^{d_{\boldsymbol{y}}})$. Moreover, since $C_4 = \operatorname{ess\,sup}_{\boldsymbol{z}\in\mathbb{R}^{d_r}}\|\nabla_{\boldsymbol{\theta}}\mathbf{T}(\boldsymbol{z})\| < \infty$, we have

$$C_2 \leq C_1C_4\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\|\nabla\mathbf{g}(\boldsymbol{z})\|^2\right] < \infty, \quad C_3 \leq C_1C_4\mathbb{E}_{(\boldsymbol{y},\boldsymbol{z})\sim\gamma\otimes\pi}\left[\left\|\mathbf{g}_{\mathrm{opt}}(\boldsymbol{z}) - \boldsymbol{V}^*\boldsymbol{y}\right\|^2\right] < \infty.$$

Therefore, we have

$$\begin{aligned}
\left(\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\left\|\nabla_{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}\widetilde{\mathcal{L}}^{\boldsymbol{y}}(\boldsymbol{\theta})\right\|\right]\right)^2 &\leq \max\{C_2,C_3\}C_1C_4\\
&\quad\times\left(\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\mathbf{g}_{\mathrm{opt}}(\boldsymbol{z}) - \mathbf{g}(\boldsymbol{z})\right\|^2 + \left\|\nabla\mathbf{g}_{\mathrm{opt}}(\boldsymbol{z}) - \nabla\mathbf{g}(\boldsymbol{z})\right\|_F^2\right]\right).
\end{aligned}$$

□

Proof of Corollary 3.3. By the Polyak–Lojasiewicz inequality and the results in Part I, we have

$$\begin{aligned}\mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\sqrt{\mathcal{L}^{\boldsymbol{y}}(\widetilde{\boldsymbol{\theta}}^{\boldsymbol{y},\dagger})-\mathcal{L}^{\boldsymbol{y}}(\boldsymbol{\theta}^{\boldsymbol{y},\dagger})}\right] &\leq \mathbb{E}_{\boldsymbol{y}\sim\gamma}\left[\frac{1}{\sqrt{2\lambda^{\boldsymbol{y}}}}\left\|\nabla_{\boldsymbol{\theta}}\mathcal{L}(\widetilde{\boldsymbol{\theta}}^{\boldsymbol{y},\dagger})\right\|\right]\\ &\leq \frac{1}{C_5}\max\{C_2,C_3\}C_1C_4\\ &\quad\times\left(\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\left\|\mathbf{g}_{\mathrm{opt}}(\boldsymbol{z})-\mathbf{g}(\boldsymbol{z})\right\|^2+\left\|\nabla\mathbf{g}_{\mathrm{opt}}(\boldsymbol{z})-\nabla\mathbf{g}(\boldsymbol{z})\right\|_F^2\right]\right)^{1/2},\end{aligned}$$

where $C_5=\operatorname{ess\,inf}_{\boldsymbol{y}\sim\gamma}\sqrt{2\lambda^{\boldsymbol{y}}}>0$. □

Appendix E. Detailed Discussion of LazyDINO algorithm

In this section, we describe algorithms for performing inference with a `LazyDINO` algorithm, comprised of an offline and two online phases. In Appendix E.1, we describe the *offline* surrogate construction phase, that is, computing the encoder and decoder for the parameter latent space and then training `rb-DINO` in that latent space. Next, in Appendix E.2, we describe the *online* solving of a BIP for a given observed data vector $\boldsymbol{y}$ using LMVI. Lastly, in **??**, we discuss the *amortization* that the `LazyDINO` algorithm achieves in contrast to model-based LMVI (`LazyMap`)and also compare it to simulation-based AVI.

Appendix E.1. Offline phase: surrogate construction

Algorithm 1: `LazyDINO`: define latent space and embed dataset

Input:
(i) prior distribution sampler: μ, prior precision operator: $\mathcal{C}^{-1}$
(ii) noise precision matrix: Γ_n^{-1}, observable basis $\boldsymbol{V}$
(iii) PtO map: $m\mapsto\boldsymbol{\mathcal{G}}(m)$, Jacobian action: $D\boldsymbol{\mathcal{G}}(m)$
(iv) # desired training dataset samples: $N\in\mathbb{N}$
(v) # samples to compute encoder/decoder: $N_L\leq N$
(vi) embedding dimension: d_r or eigenvalue tail sum tolerance: ϵ_L

Output:
(i) encoder/decoder pair: $\mathcal{E}_r,\mathcal{D}_r$
(ii) latent space training dataset inputs: $\{\boldsymbol{z}^{(j)}\}$, outputs: $\{\boldsymbol{g}^{(j)}\}$, $\{\boldsymbol{J}_r^{(j)}\}$, $j=1,\dots,N$

begin
1. $m^{(j)}\sim\mu,\quad j=1,\dots,N$ ▷ `Sample prior`
2. $\boldsymbol{\mathcal{G}}(m^{(j)}),D_H\boldsymbol{\mathcal{G}}(m^{(j)}),\quad j=1,\dots,N$ ▷ `Evaluate PtO map/Jacobian`
3. `Create encoder/decoder:`
   $\{\psi_k\in\mathscr{M}\}_{k=1}^{d_r}\leftarrow$ `eigenvalue_problem`$\left(\left\{D\boldsymbol{\mathcal{G}}(m^{(j)})\right\}_{i=1}^{N_L},\Gamma_n^{-1},\mathcal{C}^{-1},\epsilon_L \text{ or } d_r\right)$ ▷ (32), (E.1)
   $\mathcal{D}_r\boldsymbol{z}:=\sum_{k=1}^{d_r}\boldsymbol{z}_k\psi_k,\ \mathcal{E}_r:=\mathcal{D}_r^{\top}\mathcal{C}^{-1}$ ▷ (15), (17)
4. `Embed dataset:` ▷ (38), (42)
   $\boldsymbol{z}^{(j)}\leftarrow\mathcal{E}_r m^{(j)},\ \boldsymbol{g}^{(j)}\leftarrow\boldsymbol{V}^{\top}\Gamma_n^{-1}\boldsymbol{\mathcal{G}}(m^{(j)}),\ \boldsymbol{J}_r^{(j)}\leftarrow\boldsymbol{V}^{\top}\Gamma_n^{-1}D\boldsymbol{\mathcal{G}}(m^{(j)})\mathcal{D}_r,\quad j=1,\dots,N$

end

Latent space: solving the generalized eigenvalue problem. We describe here the computation of the `eigenvalue_problem` in Algorithm 1 to find the reduced basis Ψ_r for our encoder and decoder. Motivated by Theorem 3.1, we take Ψ_r to be composed of the leading $H_{\mathcal{C}}$-orthonormal eigenbasis functions of an MC approximation of the generalized eigenvalue problem defined in (32):

$$\mathcal{C}^{-1}\mathcal{H}_A\approx\frac{1}{N_L}\sum_{j=1}^{N_L}D\boldsymbol{\mathcal{G}}(m^{(j)})^{*}\Gamma_n^{-1}D\boldsymbol{\mathcal{G}}(m^{(j)}),\quad m^{(j)}\overset{\text{i.i.d.}}{\sim}\mu. \tag{E.1}$$

Following (33), the latent parameter space dimension $d_r \leq \dim(\mathscr{M})$ is chosen to capture the dominant information in $\mathcal{H}_{\mathcal{A}}$; specifically, it is desirable to ensure that the eigenvalue tail sum is small. For many high- to infinite-dimensional BIPs, d_r is expected to be small due to, e.g., Saint-Venant's principle for coercive elliptic PDEs, concentration of measure, and the low-rankness of sparse observations extracted from a PDE state.

When the discretization dimension of the problem is large, the generalized eigenvalue decomposition must be computed matrix-free. To this end there are many suitable computational tools such as randomized methods [106–108] and Krylov subspace methods [109–113]. In this case, since the computation of the eigenvalue tail is intractable, one can adaptively find a sufficiently large dimension, d_r, such that the eigenvalues have decayed sufficiently, i.e., λ_{d_r}/λ_1 is small.

The samples needed to compute the reduced basis can be reused as part of the training data set; far fewer samples are usually needed to compute the reduced basis than are needed to train `RB-DINO` to required low error tolerances, see e.g. [29]. In this case, the additional training sample latent Jacobians can be directly formed via its action or adjoint action, as in (42). Specifically, once the PtO evaluation at $m^{(j)}$ is available, only $\min(d_{\boldsymbol{y}}, d_r)$ evaluations of the PtO map derivative or its adjoint action are needed for a latent Jacobian evaluation. The evaluation cost can often be reduced to a fraction of the PtO map evaluation cost for linear or highly nonlinear PDEs. For details on efficient means to form latent Jacobian matrices for PDE-constrained PtO maps, see Appendix F and [29, Section 4.3]. Empirical evidence of the low relative cost of Jacobians can be seen in our numerical results in Section 6.3 in Table 3 and Table 4.

Using the encoder and decoder, defined previously in terms of the reduced basis Ψ_r and prior precision $\mathcal{C}^{-1}$ in (15) and (17), we embed the training data into the latent space, resulting in the whitened latent inputs $\boldsymbol{z}^{(j)}$, whitened PtO samples $\boldsymbol{g}^{(j)}$, and whitened latent Jacobian samples $\boldsymbol{J}_r^{(j)}$. A summary of these procedures is given in Algorithm 1.

RB-DINO training. Next, in Algorithm 2, we train `RB-DINO` using the embedded data set. This involves a straightforward empirical risk minimization arising from MC estimate of either (37) or (41). Any method for stochastic unconstrained optimization, e.g., stochastic gradient descent, Adam [97], or second-order methods [114, 115] can be used. If the conventional L^2_μ empirical risk is employed, we refer to the surrogate as `RB-NO` (neural operator) instead of `RB-DINO` (derivative-informed neural operator).

Algorithm 2: `LazyDINO`: train reduced basis neural operator in latent space $\mathbb{R}^{d_r}$

Input:

(i) training dataset inputs: $\{\boldsymbol{z}^{(j)}\}$, outputs: $\{\boldsymbol{g}^{(j)}\}, \{\boldsymbol{J}_r^{(j)}\}$, $j = 1, \ldots, N$

(ii) untrained neural network: $\mathbf{g}_w : \mathbb{R}^{d_r} \times \mathbb{R}^{d_W} \to \mathbb{R}^{d_{\boldsymbol{y}}}$

(iii) choice of conventional L^2_μ (`RB-NO`) or derivative-informed H^1_μ (`RB-DINO`) objective

Output:

(i) trained neural network: $\mathbf{g}_{\boldsymbol{w}^*}$

begin

`1.` Train $\mathbf{g}_w$ by minimizing an empirical risk:

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{w}\in\mathbb{R}^{d_W}} \frac{1}{N}\sum_{j=1}^{N} \left(\left\|\boldsymbol{g}^{(j)} - \mathbf{g}_w\left(\boldsymbol{z}^{(j)}\right)\right\|^2 + \underbrace{\left\|\boldsymbol{J}_r^{(j)} - \nabla_{\boldsymbol{z}}\mathbf{g}_w\left(\boldsymbol{z}^{(j)}\right)\right\|_F^2}_{\text{include for } H^1_\mu \text{ (RB-DINO) objective}} \right)$$

end

Equipped with a sufficiently accurate neural network approximation to the optimal latent PtO map $\mathbf{g}_{\text{opt}}$, including accurate approximations of its derivatives, one can proceed to transport map variational inference.

Appendix E.2. Online phase: lazy map variational inference with surrogate latent objective function

We perform LMVI using a stochastic approximation of the surrogate rKL objective and its gradient defined in (43a). Algorithm 3 summarizes the `LazyDINO` training procedure when using first order methods.

Algorithm 3: `LazyDINO`: train transport map w/surrogate objective function in latent space $\mathbb{R}^{d_r}$

Input:
(i) whitened latent prior sampler: π
(ii) single-sample surrogate latent space rKL objective for observation $\boldsymbol{y}$: $\widetilde{\mathcal{L}}^{\boldsymbol{y}}_{1,r}(\cdot,\cdot\,;\boldsymbol{w}^*)$
(iii) untrained transport map with random initial weights: $\mathsf{T}_{\boldsymbol{\theta}} : \mathbb{R}^{d_r} \to \mathbb{R}^{d_r}$, $\boldsymbol{\theta}^0$
(iv) J batch sizes, learning rates, # iterations: $(B_j, a_j, I_j), j = 1, \ldots, J$

Output:
(i) trained transport map with pushforward density: $\mathsf{T}_{\boldsymbol{\theta}^*}$, $(\mathsf{T}_{\boldsymbol{\theta}^*})_\sharp \pi$

begin
  for $j = 1, \ldots, J$ **do**
    for $i = 1, \ldots, I_j$ **do**
      $\left\{\boldsymbol{z}^{(k)}\right\}_{k=1}^{B_j} \sim \pi$ ▷ `Sample a new stochastic batch`
      $\Delta\boldsymbol{\theta}^i \leftarrow \frac{1}{B_j}\sum_{k=1}^{B_j} \nabla_{\boldsymbol{\theta}} \widetilde{\mathcal{L}}^{\boldsymbol{y}}_{1,r}(\boldsymbol{z}^{(k)}, \boldsymbol{\theta}^{i-1}; \boldsymbol{w}^*)$ ▷ `Estimate gradient of objective function`
      $\boldsymbol{\theta}^i \leftarrow$`stochastic_gradient_based_iteration`$(\Delta\boldsymbol{\theta}^i, (\boldsymbol{\theta}^0, \ldots, \boldsymbol{\theta}^{i-1}), \alpha_j)$ ▷ `e.g., Adamax`
    end
    $\boldsymbol{\theta}^0 \leftarrow \boldsymbol{\theta}^{I_j}$
  end
  $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^{I_J}$ ▷ `Last parameter is approximately optimal`
end

Since our latent PtO surrogate is a neural network, an MC estimate of the gradient of the rKL objective with respect to transport map parameters, i.e., $\mathbb{E}_{\boldsymbol{z}\sim\pi}\left[\widetilde{\mathcal{L}}^{\boldsymbol{y}}_{1,r}(\boldsymbol{z},\boldsymbol{\theta};\boldsymbol{w}^*)\right]$, can be computed rapidly on GPUs, especially when the surrogate objective function gradient is a compiled batch-vectorized expression. In practice, for each example we study (in Section 5.1, Section 5.2), the evaluation time of the surrogate objective function gradient is orders of magnitude less than the evaluation time of the original PtO map-dependent `LazyMap` objective gradient.

Since iterations can be performed rapidly, we use rounds of stochastic approximation-based (SA) optimization, increasing the batch sample size [116] and decreasing the learning rate each time for a number of iterations that is computationally tractable. We found that using such a strategy was more successful than using either small or large batch sizes alone. This strategy is similar to *retrospective approximation* (RA) [117].

Appendix F. Forming the reduced Jacobian through direct and adjoint sensitivities

While our discussion in the main body targets general BIPs, an important class is BIPs constrained by PDE models. This section mentions a few points regarding this class of problems. We consider a nonlinear variational residual problem involving an additional state variable $u \in \mathscr{U}$. The PtO map can then be written abstractly as

$$\boldsymbol{\mathcal{G}}(m) : m \mapsto u \mapsto \boldsymbol{\mathcal{O}}(u) \qquad \text{such that} \qquad \mathcal{R}(u, m) = 0 \in \mathscr{U}', \tag{F.1}$$

where $\boldsymbol{\mathcal{O}} : \mathscr{U} \to \mathbb{R}^{d_{\boldsymbol{y}}}$ is an observation operator, $\mathcal{R} : \mathscr{U} \times \mathscr{M} \to \mathscr{U}'$ is the residual of the PDE model, and $\mathscr{U}'$ is the topological dual of $\mathscr{U}$.

Derivatives of $\boldsymbol{\mathcal{G}}$ with respect to m require implicit differentiation through the residual equation. They are well-defined if the conditions of the implicit function theorem are met (e.g., isolated solution, regular branch of solutions, stability, and sufficient resolution of the discretization of the PDEs).

At a given sample point, d_r derivative action or $d_{\boldsymbol{y}}$ derivative adjoint actions are required to form the latent Jacobian in (42). In particular, the derivative can be expressed as

$$D\boldsymbol{\mathcal{G}}(m) = -D\boldsymbol{\mathcal{O}}(u)\left[\partial_u \mathcal{R}(u, m)\right]^{-1} \partial_m \mathcal{R}(u, m). \tag{F.2}$$

The dominant cost is in the inverse actions of $\partial_u \mathcal{R}(u,m)$ or $\partial_u \mathcal{R}(u,m)^*$ on the parameter or observable basis, where each action requires solving a linear PDE. This cost can be considerably reduced when sparse direct solvers are used, as one can amortize the factorization costs associated with these actions on all parameter or observable bases. The cost reduction is significant for large-scale linear PDE models and highly nonlinear PDE models.

Appendix G. Defining the Laplace approximation baseline and its computation

The Laplace approximation has a long history rooted in the work of Laplace (1774) [33] and is especially important for approximate Bayesian inversion in high dimensions. Efficient estimates of the Laplace Approximation can be computed for problems that exhibit informativeness only in a parameter subspace.

We define the Laplace Approximation as the Gaussian distribution centered at the unique strong minimizer, the *maximum-a-posteriori* (MAP) estimate m_{MAP}, of the Onsager-Machlup functional $I_{\mu^{\boldsymbol{y}}} : \mathscr{M} \to \mathbb{R}^+$ of $\mu^{\boldsymbol{y}}$, assuming it exists (see [118]), with covariance defined as the inverse of the Hessian operator of the functional at the MAP estimate. For the posteriors considered in this work, the Onsager-Machlup functional is the sum of the potential function $\Phi^{\boldsymbol{y}}$ and the Onsager-Machlup functional of the Gaussian prior distribution, i.e.

$$\mu_{\text{LA}} = \mathcal{N}(m_{\text{MAP}}, \mathcal{C}_{\text{LA}}), \quad \begin{cases} m_{\text{MAP}} = \operatorname*{argmin}_{m \in \mathscr{M}} I_{\mu^{\boldsymbol{y}}}(m), \\ I_{\mu^{\boldsymbol{y}}}(m) = \Phi^{\boldsymbol{y}}(m) + \frac{1}{2}\|m\|^2_{\mathcal{C}^{-1}}, \\ \mathcal{C}_{\text{LA}} = (D^2 I(m_{\text{MAP}}))^{-1}, \end{cases} \tag{G.1}$$

so long as the potential function is Lipschitz continuous. The Onsager-Machlup functional $I_{\mu^{\boldsymbol{y}}}$ generalizes the commonly known *negative log-posterior density* with respect to Lebesgue measure, $\log \pi^{\boldsymbol{y}}$, to posterior probability distributions, see e.g. [118] for more.

In our numerical examples, we use an efficient Inexact Newton–Conjugate Gradients numerical optimization algorithm [119, 120] to find the MAP estimate, m_{MAP} which converged usually within $O(10) - O(100)$ inexact Newton iterations. This is conservatively estimated to be equivalent in cost to 100 evaluations of the PtO map in the results section.

Appendix H. On estimating density-based diagnostics

The key to computing density-based diagnostics is to evaluate the Radon–Nikodym derivative between the posterior approximation of interest and the prior, i.e., the approximate likelihood evaluations. Here, we provide the formula for this Radon–Nikodym derivative for Laplace approximation and transport map pushforward distributions.

Appendix H.1. Laplace approximation formulae

We consider the following decomposition of the LA covariance

$$\mathcal{C}_{\text{LA}} = \mathcal{C} - \mathcal{D}_{\text{LA}} \left(\frac{\lambda_j}{\lambda_j + 1} \delta_{jk} \right) \mathcal{E}_{\text{LA}} \mathcal{C}, \quad \mathcal{C}_{\text{LA}}^{-1} = \mathcal{C}^{-1} + \mathcal{C}^{-1} \mathcal{D}_{\text{LA}} (\lambda_j \delta_{jk}) \mathcal{E}_{\text{LA}},$$

where $\mathcal{D}_{\text{LA}}$ and $\mathcal{E}_{\text{LA}}$ are the linear encoder and decoder based on the eigendecomposition of the prior-preconditioned Hessian of the potential at the MAP point m_{MAP}, and $\boldsymbol{\Lambda}_{\text{LA}} = \lambda_j \delta_{ij}$ is a diagonal matrix consisting of eigenvalues. The Radon–Nikodym derivative between μ_{LA} and the prior μ is given by

$$\begin{aligned} \frac{\mathrm{d}\mu_{\text{LA}}}{\mathrm{d}\mu}(m) &= \frac{\mathrm{d}\mu_{\text{LA}}}{\mathrm{d}\mathcal{N}(0, \mathcal{C}_{\text{LA}})} \times \frac{\mathrm{d}\mathcal{N}(0, \mathcal{C}_{\text{LA}})}{\mathrm{d}\mu} \\ &= \exp\Big(-\frac{1}{2}\|m_{\text{MAP}}\|^2_{\mathcal{C}^{-1}} - \frac{1}{2}\|\mathcal{E}_{\text{LA}} m_{\text{MAP}}\|^2_{\boldsymbol{\Lambda}_{\text{LA}}} + (\mathcal{E}_{\text{LA}} m_{\text{MAP}})^\top \boldsymbol{\Lambda}_{\text{LA}} (\mathcal{E}_{\text{LA}} m) \\ &\quad + \langle m_{\text{MAP}}, m \rangle_{\mathcal{C}^{-1}} + \frac{1}{2}\sum_j \log(1+\lambda_j) - \frac{1}{2}\|\mathcal{E}_{\text{LA}} m\|^2_{\boldsymbol{\Lambda}_{\text{LA}}} \Big). \end{aligned} \tag{H.1}$$

For example, the rKL between the Laplace approximation and the true posterior is given by

$$\begin{aligned}\mathcal{D}_{\mathrm{KL}}(\mu_{\mathrm{LA}}||\mu^{\boldsymbol{y}}) &= \mathbb{E}_{m\sim\mu_{\mathrm{LA}}}\left[\log\left(\frac{\mathrm{d}\mu_{\mathrm{LA}}}{\mathrm{d}\mu}(m)\frac{\mathrm{d}\mu}{\mathrm{d}\mu^{\boldsymbol{y}}}(m)\right)\right]\\ &= \mathbb{E}_{m\sim\mu_{\mathrm{LA}}}\left[\Phi^{\boldsymbol{y}}(m)+\log\left(\frac{\mathrm{d}\mu_{\mathrm{LA}}}{\mathrm{d}\mu}(m)\right)\right]+\log Z^{\boldsymbol{y}},\end{aligned}$$

where the Radon–Nikodym derivative at parameters samples can be computed using (H.1).

Appendix H.2. lazy map pushforward posterior formulae

Let $\mathcal{T}$ be a lazy map, then we have

$$\frac{\mathrm{d}\mathcal{T}_\sharp\mu}{\mathrm{d}\mu}(m) = \frac{\mathbf{T}_\sharp\pi(\mathcal{E}_r m)}{\pi(\mathcal{E}_r m)} = \frac{(\pi\circ\mathbf{T}^{-1})(\mathcal{E}_r m)|\det\nabla\mathbf{T}^{-1}(\mathcal{E}_r m)|}{\pi(\mathcal{E}_r m)}, \tag{H.2}$$

where $\mathbf{T}$ is the latent space transport map. For example, the rKL between the lazy map pushforward and the posterior is given by:

$$\begin{aligned}D_{\mathrm{KL}}(\mathcal{T}_\sharp\mu||\mu^{\boldsymbol{y}}) &= \mathbb{E}_{m\sim\mathcal{T}_\sharp\mu}\left[\log\left(\frac{\mathrm{d}\mathcal{T}_\sharp\mu}{\mathrm{d}\mu}(m)\frac{\mathrm{d}\mu}{\mathrm{d}\mu^{\boldsymbol{y}}}(m)\right)\right]\\ &= \mathbb{E}_{m\sim\mathcal{T}_\sharp\mu}\left[\Phi^{\boldsymbol{y}}(m)+\log\left(\frac{\mathrm{d}\mathcal{T}_\sharp\mu}{\mathrm{d}\mu}(m)\right)\right]+\log Z^{\boldsymbol{y}},\end{aligned}$$

where the Radon–Nikodym derivative at parameters samples can be computed using (H.2).

Appendix I. Conventional Neural Operator training details

For Example I, to train the neural operator using the conventional L^2_μ objective, the learning rate α^j and epoch number E^j for each training data size N^j, reported here as (α^j, E^j, N^j) are $\Big\{(2\times10^{-4}, 1500, 125), (2\times10^{-4}, 1500, 250), \ldots (2\times10^{-4}, 1500, 8k), (2\times10^{-4}, 500, 16k)\Big\}$. For Example II, we used $\Big\{((1\times10^{-4}, 3000, 125), (1\times10^{-4}, 3000, 250), \ldots (1\times10^{-4}, 3000, 4k), (1\times10^{-4}, 5000, 8k), (5\times10^{-5}, 5000, 16k)\Big\}$. We used cross-validation on the test set to ensure training with these parameters led to similar training and generalization errors.

Appendix J. Additional numerical results

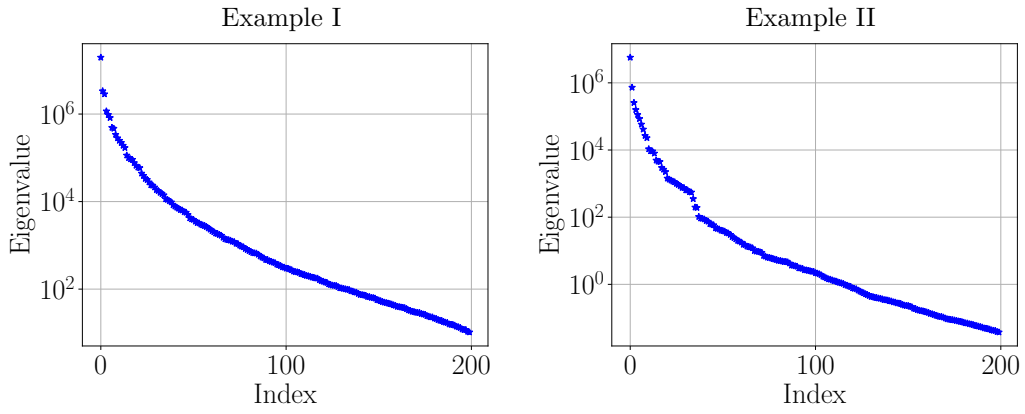

Figure J.26: Visualization of eigenvalue decay for the generalized eigenvalue problem (32) for subspace identification in the two numerical examples.

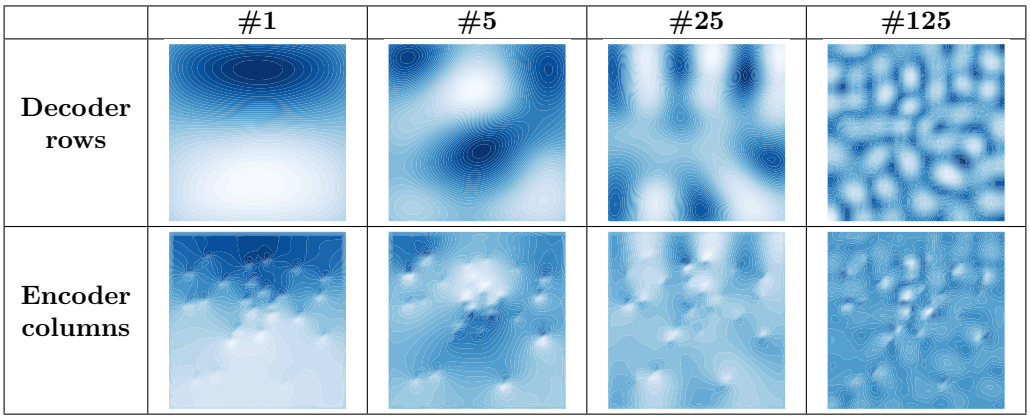

Figure J.27: **Example I.** Visualization of selected decoder rows (basis functions) and encoder columns. We note that the encoder columns are computed using the action of the prior precision operator on the decoder rows. The encoder action on input is given by the vector space inner product of the encoder columns (discretized) on the input (discretized)

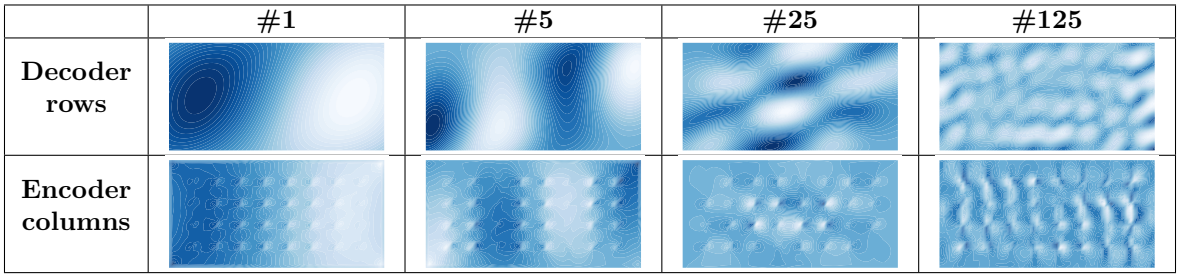

Figure J.28: **Example II.** Visualization of selected decoder rows (basis functions) and encoder columns. We note that the encoder columns are computed using the action of the prior precision operator on the decoder rows. The encoder action on an input in $\mathscr{M}$ is given by the vector space inner product of the encoder columns (discretized) on the input (discretized)

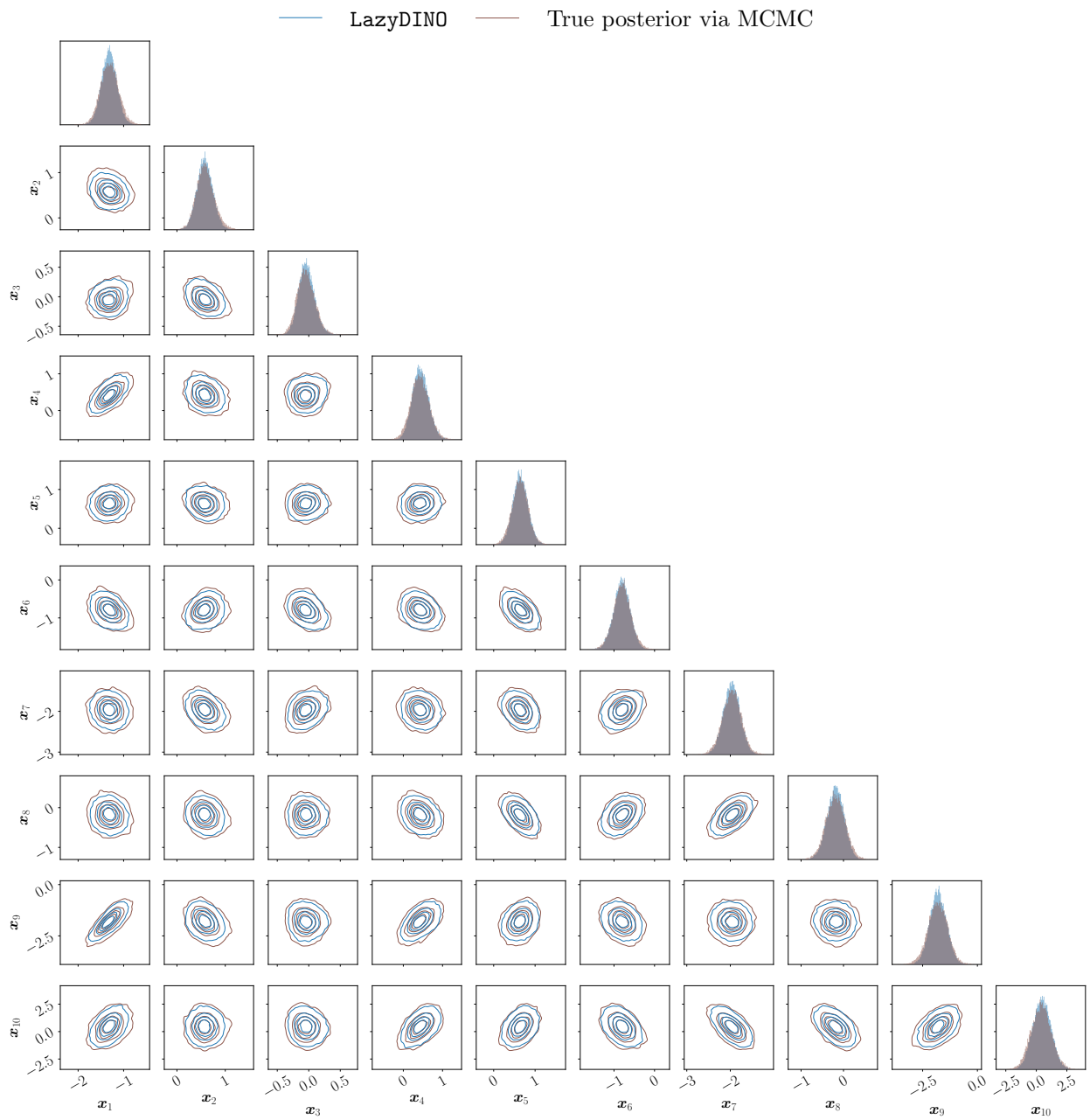

Figure J.29: **Example I BIP #1** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space.

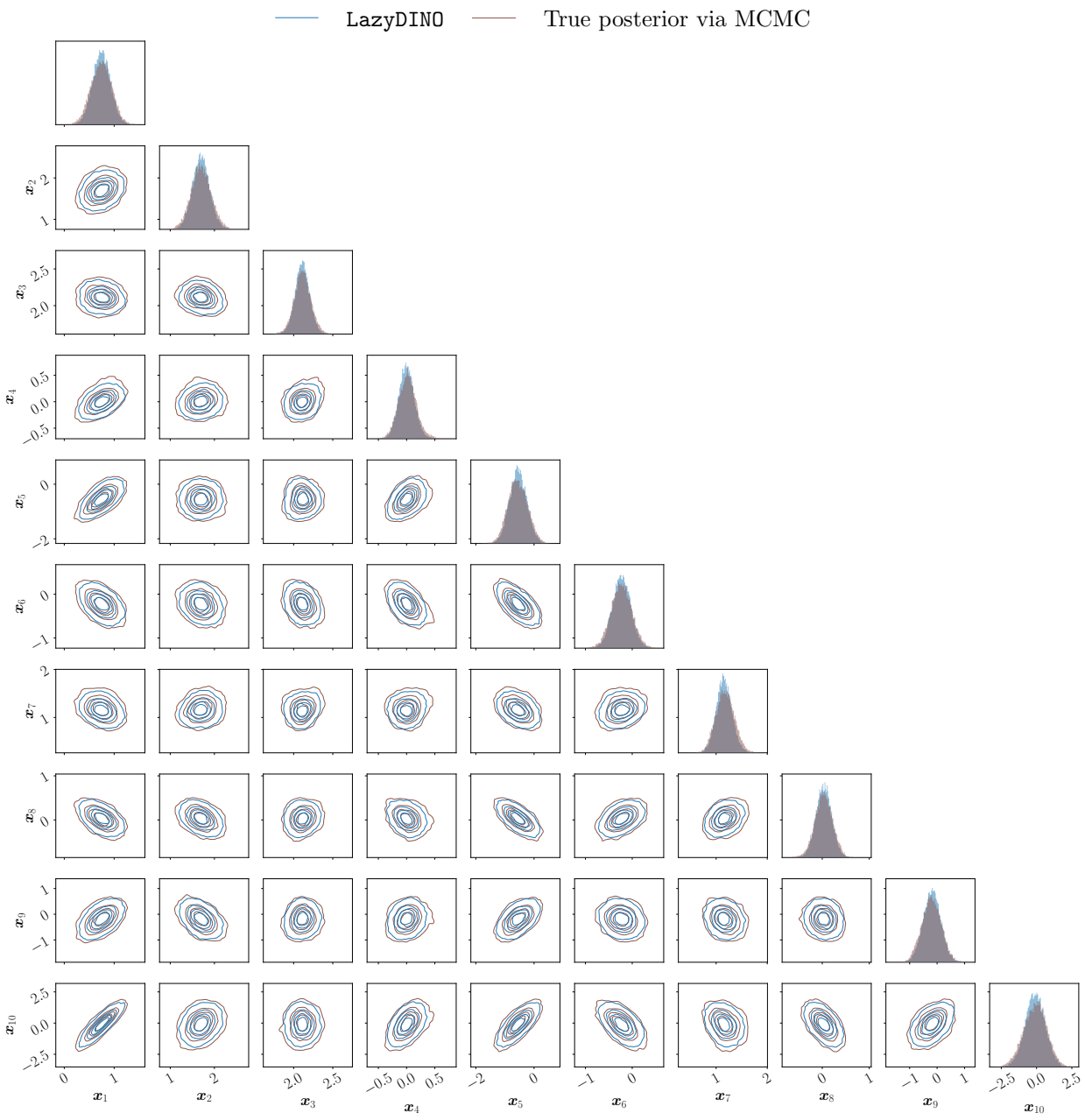

Figure J.30: **Example I BIP #2** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space.

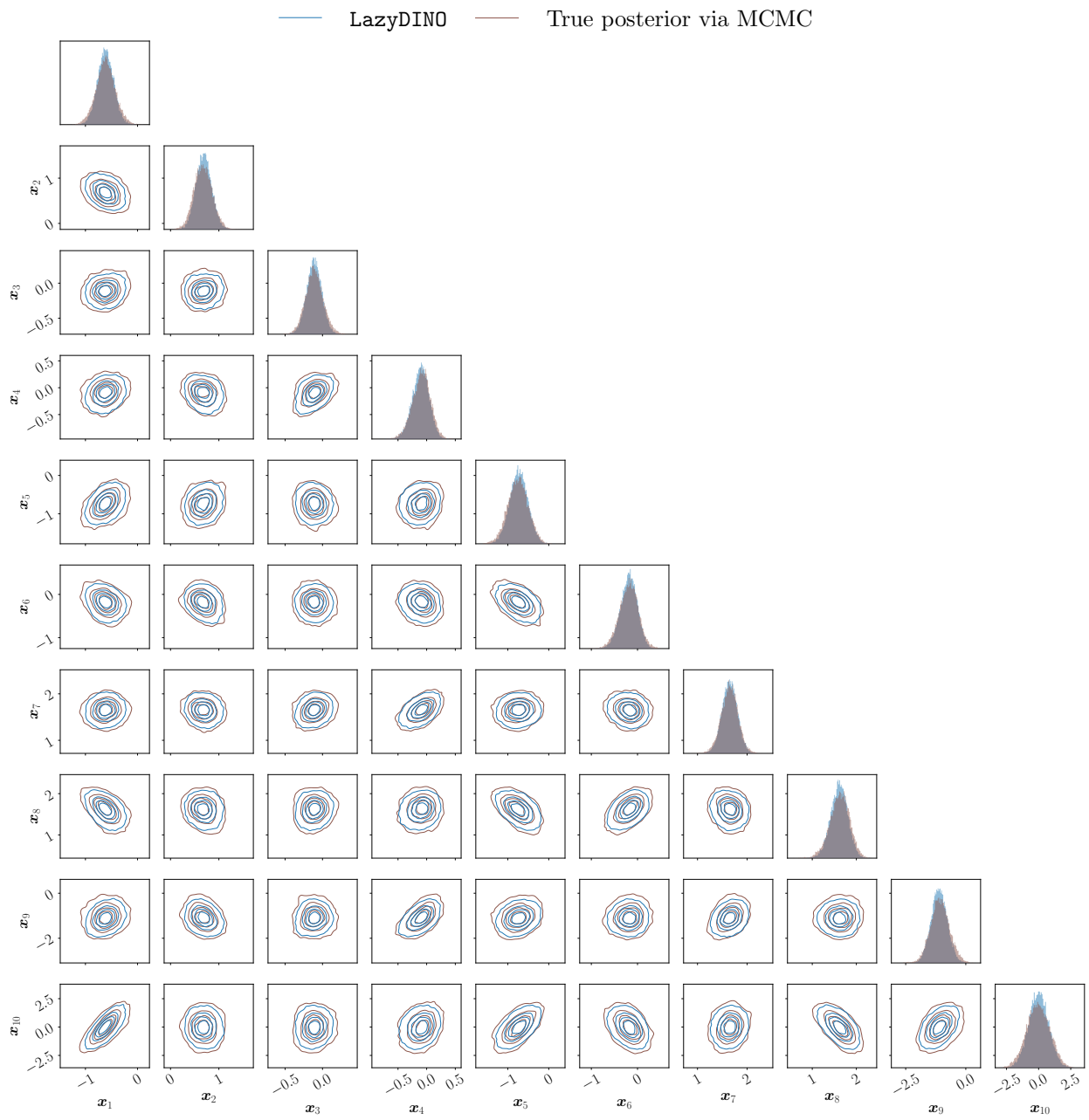

Figure J.31: **Example I BIP #3** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space.

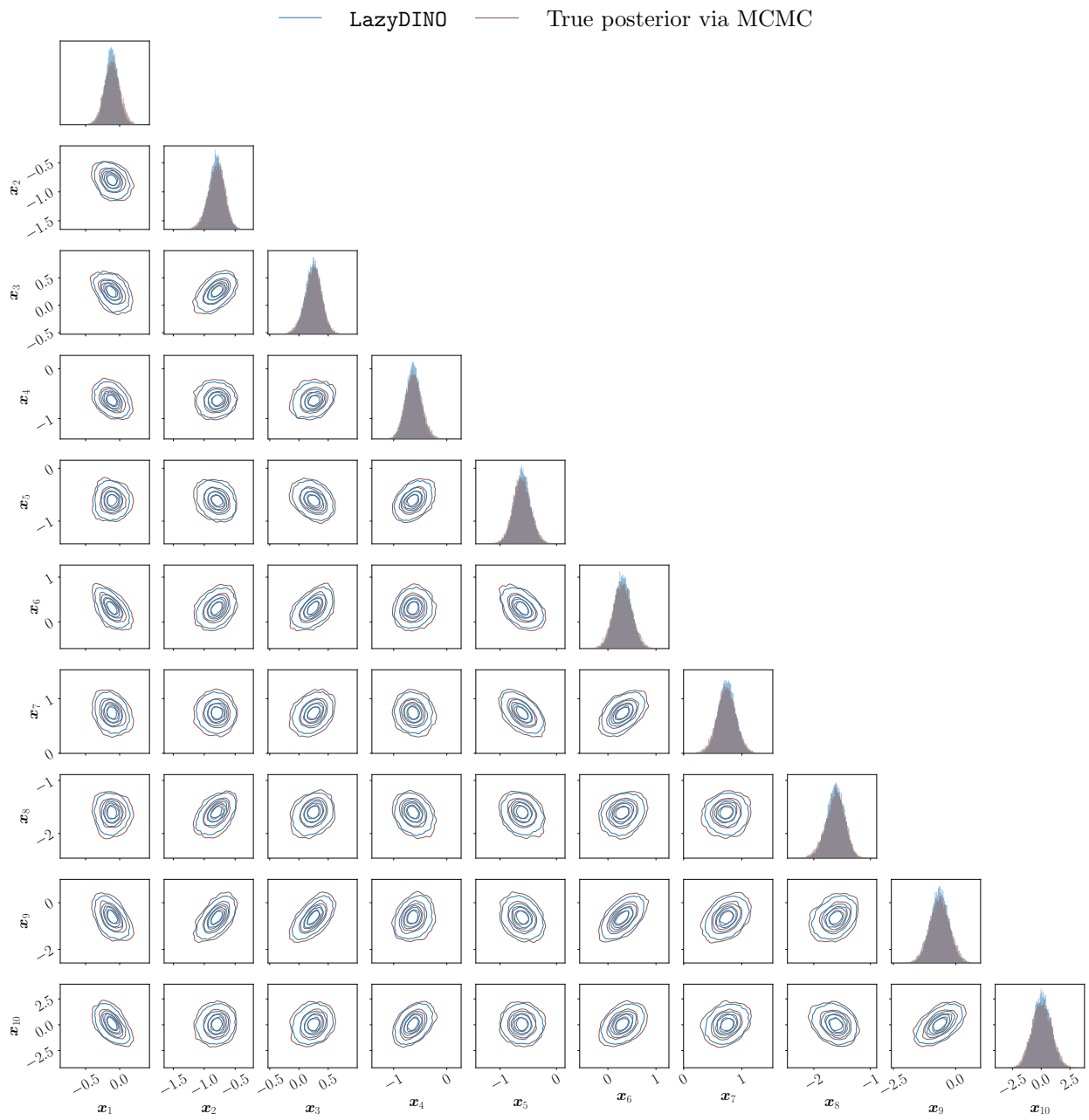

Figure J.32: **Example I BIP #4** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space.

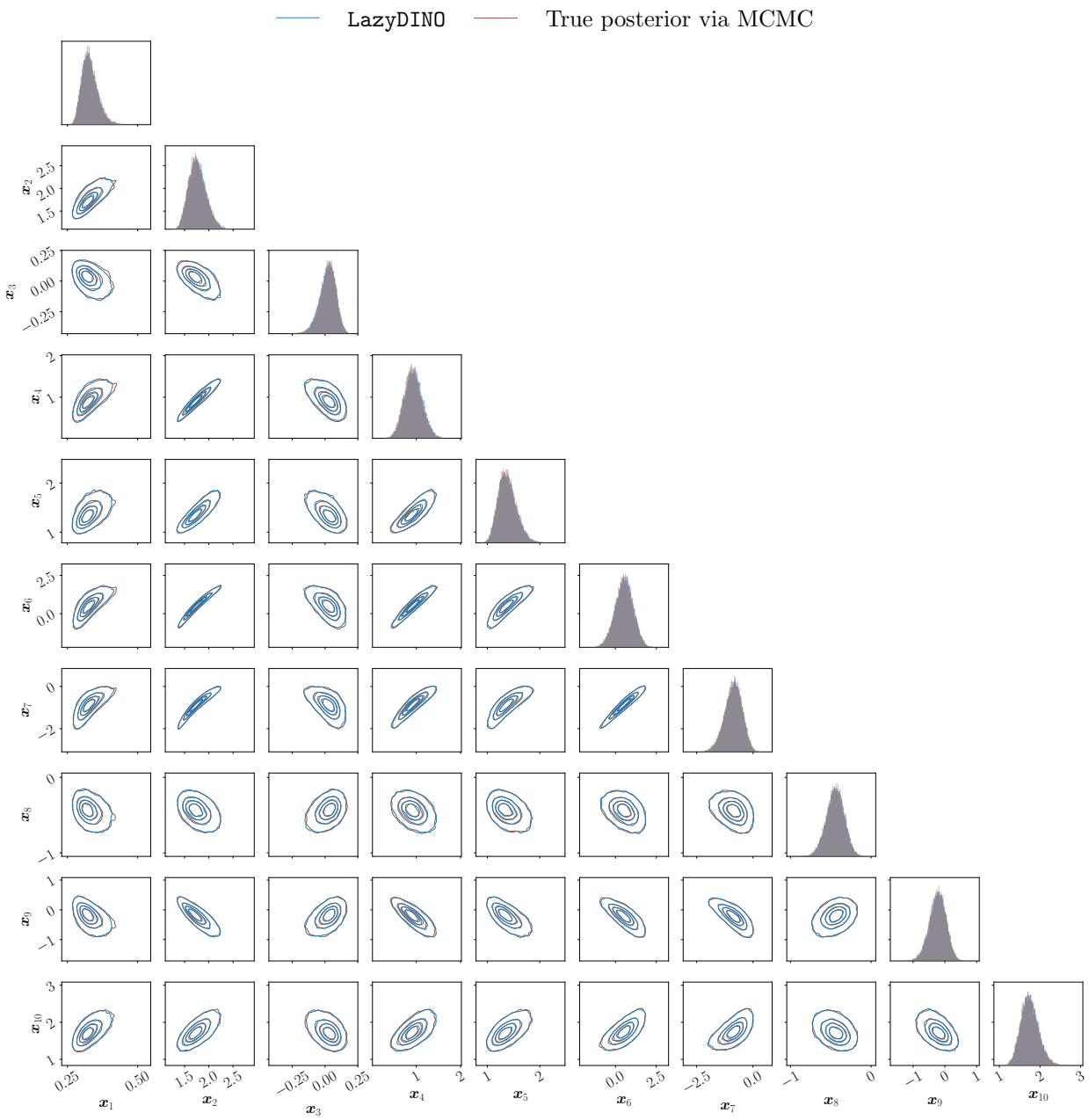

Figure J.33: **Example II BIP #1** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space.

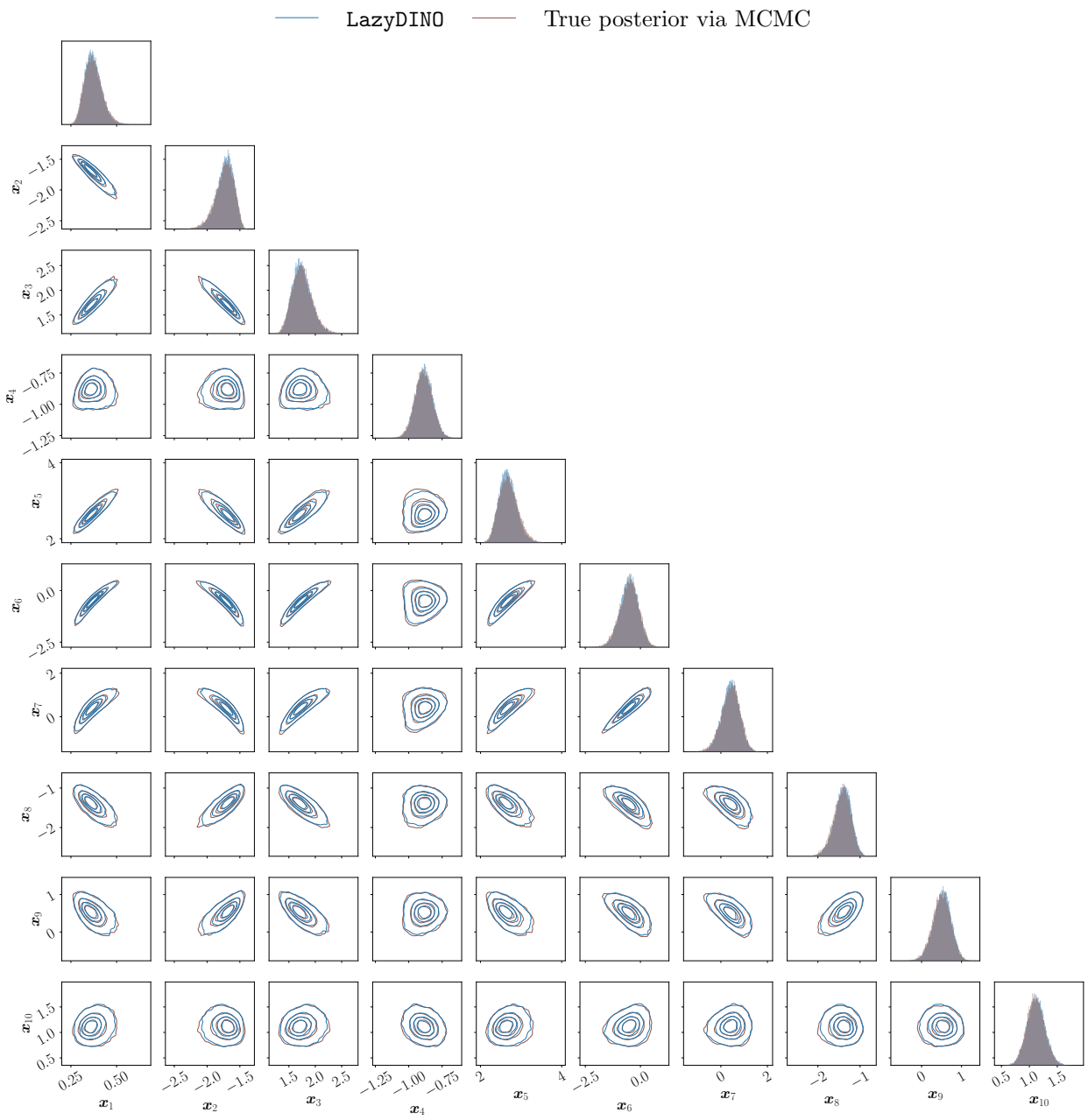

Figure J.34: **Example II BIP #2** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space in the ten leading dimensions of the latent space.

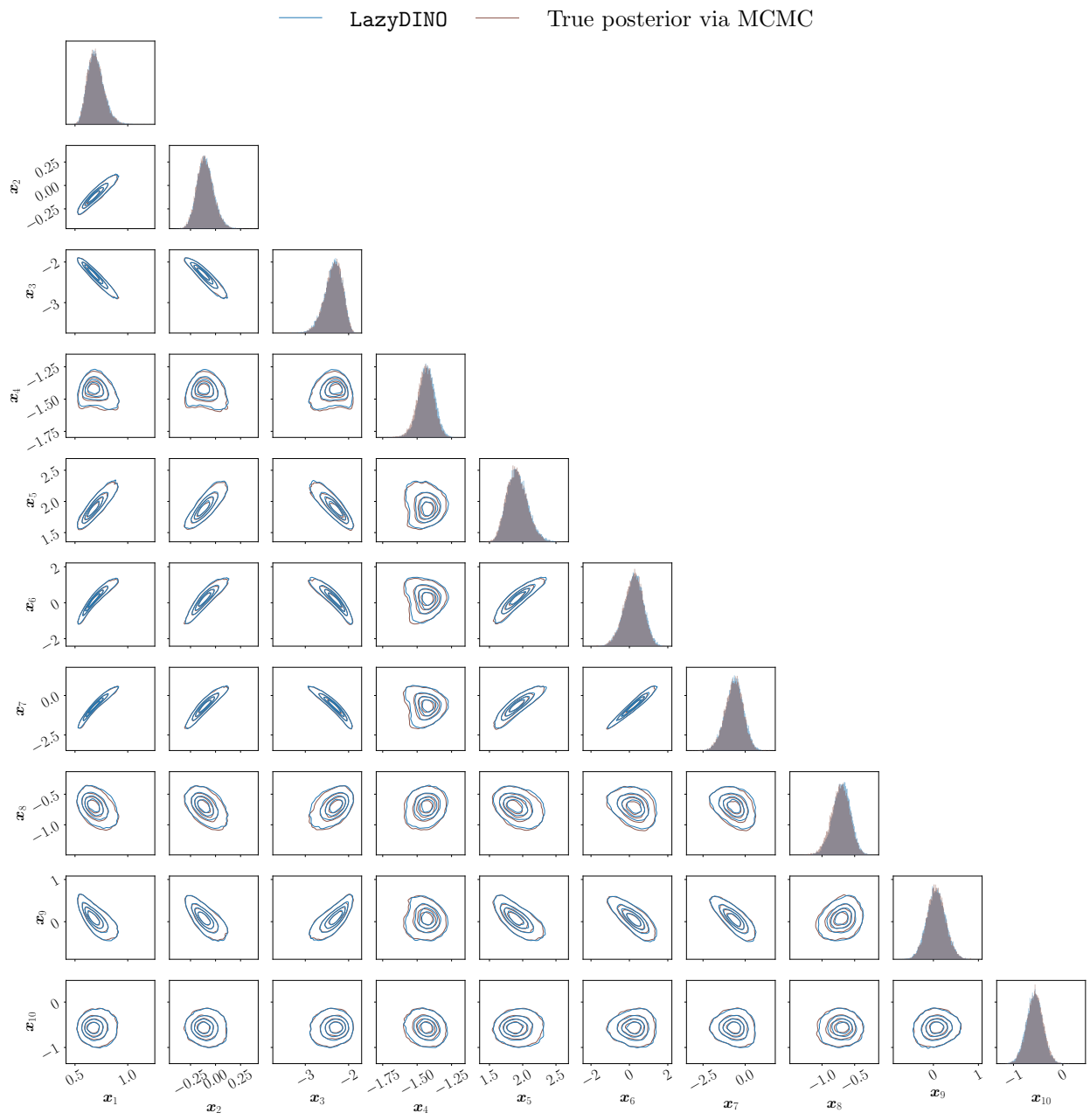

Figure J.35: **Example II BIP #3** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space.

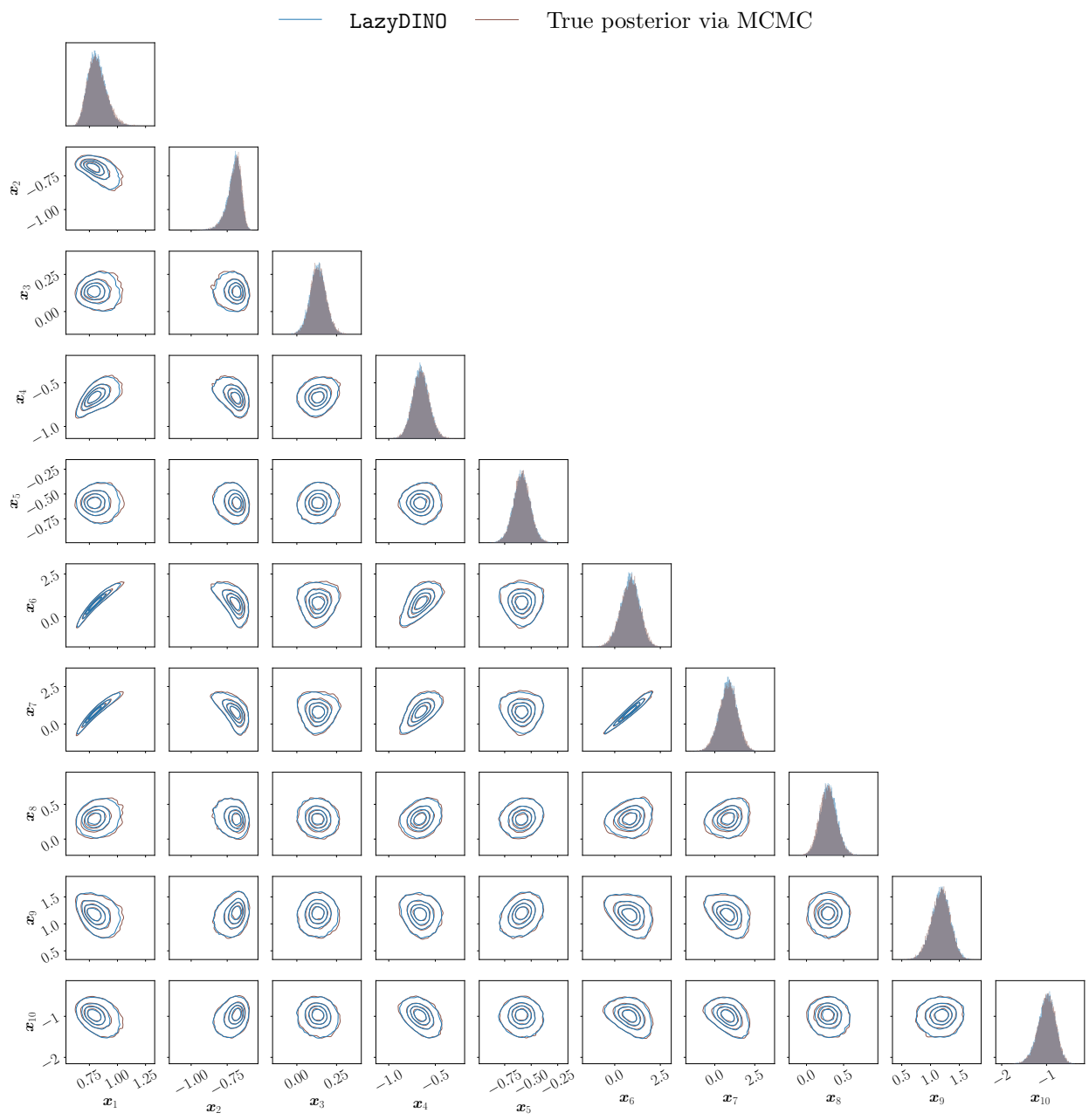

Figure J.36: **Example II BIP #4** `LazyDINO` v.s. true posterior marginals at 16k `DINO` training samples in the ten leading dimensions of the latent space.

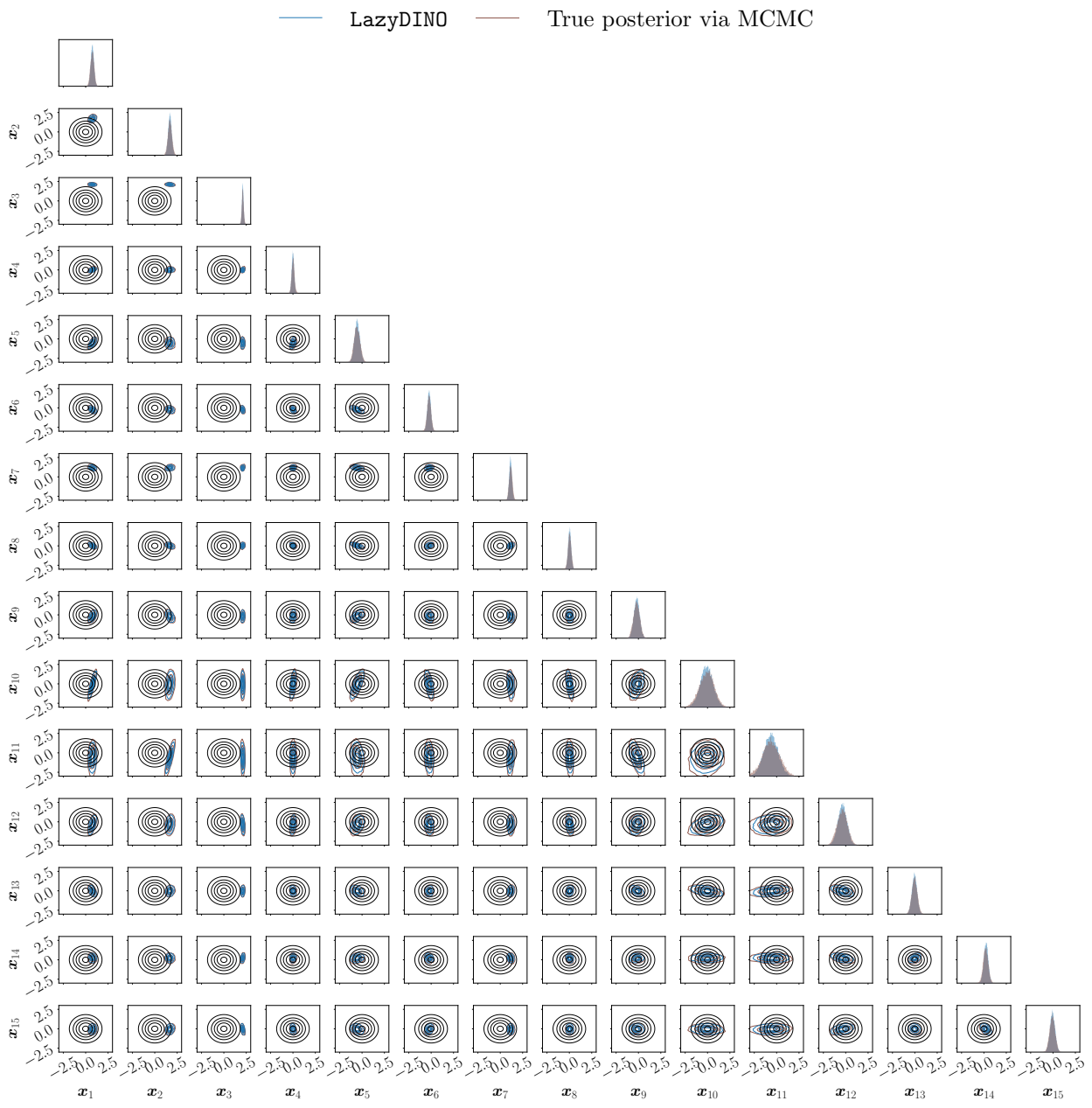

Figure J.37: **Example I BIP #2** `LazyDINO` v.s. true posterior marginals compared to prior marginals (black contour lines) at 16k `DINO` training samples in the fifteen leading dimensions of the latent space. The posterior exhibits strong concentration relative to the prior. Impressively, the offline `DINO` surrogate training strategy is able to effectively equip Bayesian inversion even with such highly concentrated posteriors.

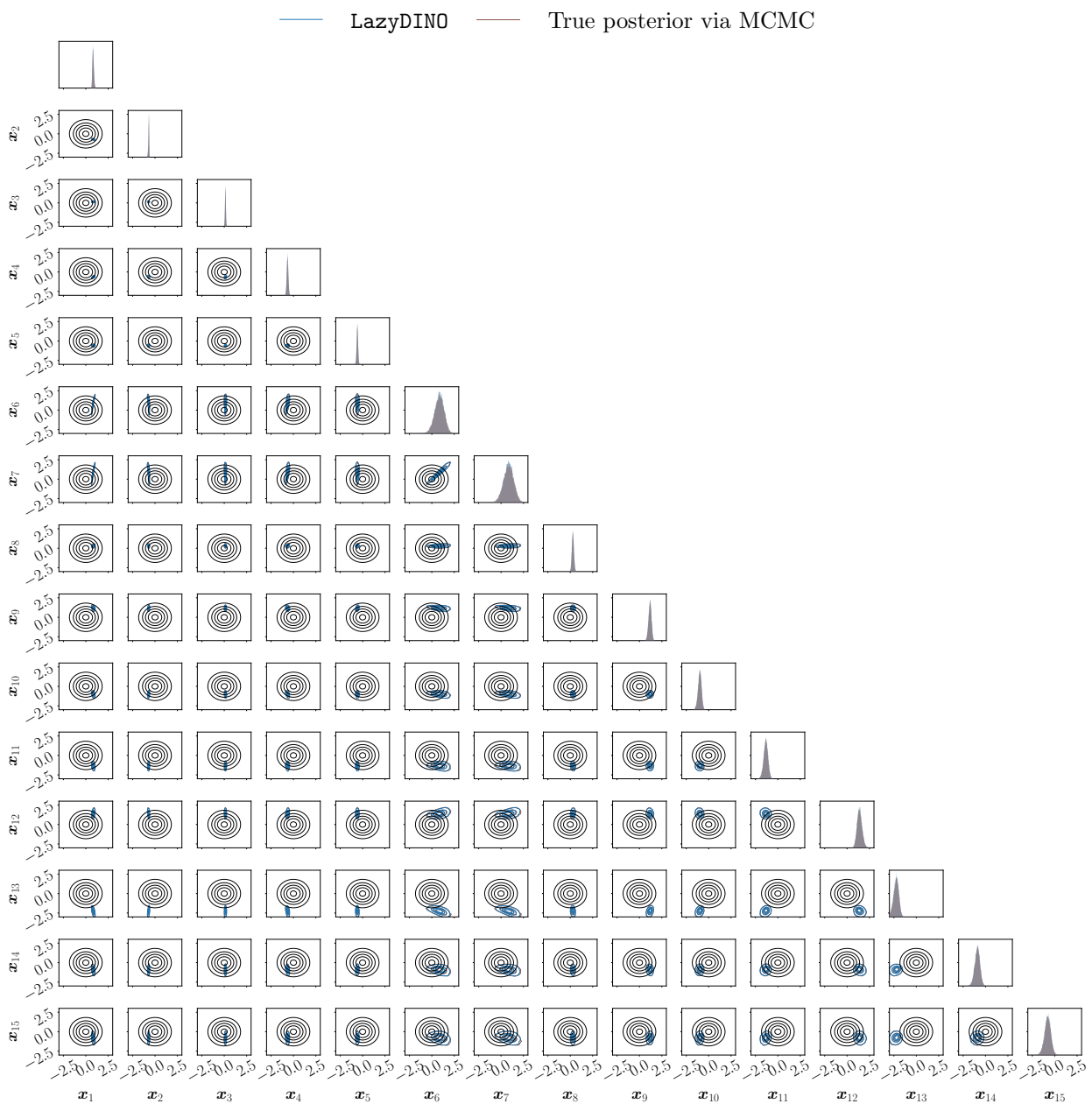

Figure J.38: **Example II BIP #4** `LazyDINO` v.s. true posterior marginals compared to prior marginals at 16k `DINO` training samples in the fifteen leading dimensions of the latent space.