Diffeomorphism-based feature learning using Poincaré inequalities on augmented input space

Romain Verdière, Clémentine Prieur, Olivier Zahm

Diffeomorphism-based feature learning using Poincaré inequalities on augmented input space

Romain Verdière romain.verdiere@inria.fr
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France

Clémentine Prieur clementine.prieur@univ-grenoble-alpes.fr
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France

Olivier Zahm olivier.zahm@inria.fr
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France

Abstract

We propose a gradient-enhanced algorithm for high-dimensional function approximation. The algorithm proceeds in two steps: firstly, we reduce the input dimension by learning the relevant input features from gradient evaluations, and secondly, we regress the function output against the pre-learned features. To ensure theoretical guarantees, we construct the feature map as the first components of a diffeomorphism, which we learn by minimizing an error bound obtained using Poincaré Inequality applied either in the input space or in the feature space. This leads to two different strategies, which we compare both theoretically and numerically and relate to existing methods in the literature. In addition, we propose a dimension augmentation trick to increase the approximation power of feature detection. A generalization to vector-valued functions demonstrate that our methodology directly applies to learning autoencoders. Here, we approximate the identity function over a given dataset by a composition of feature map (encoder) with the regression function (decoder). In practice, we construct the diffeomorphism using coupling flows, a particular class of invertible neural networks. Numerical experiments on various high-dimensional functions show that the proposed algorithm outperforms state-of-the-art competitors, especially with small datasets.

Keywords: high-dimensional function approximation, nonlinear feature learning, augmented space, Poincaré Inequality, invertible neural networks.

1 Introduction

Modern computational models for scientific and engineering applications typically involve a large number of input parameters and are expensive-to-evaluate both in time and resources. Replacing the model with an accurate and fast-to-evaluate surrogate (or approximation) offers a viable workaround in many applications. Approximating such high-dimensional functions with classical approximation tools such as polynomials, wavelets or neural networks is, however, a difficult task. This is even aggravated in the small sample regime where one only has access to a little number of model evaluations. One way to address this challenge is to reduce the input dimension beforehand. This consists in approximating the model as the composition of two functions: a *feature map* which extracts the relevant features of the

input variables, and a *profile function* which regresses the model output on the features. Manifold learning methods (Donoho and Grimes, 2003; Tenenbaum et al., 2000; Schölkopf et al., 2005), to cite just a few, aim to detect a low-dimensional nonlinear manifold on which input parameters are concentrated. These methods do not account for the model to be approximated and hence are *unsupervised* dimension reduction methods.

This paper is concerned with *supervised* dimension reduction techniques where the feature map is built using gradients of the model. Those gradients can be typically computed via automatic differentiation (Griewank et al., 1989; Baydin et al., 2018) or via adjoint-state method (Plessix, 2006) at a cost which is comparable to the cost of evaluating the model itself. Linear dimension reduction using model gradients has been firstly proposed in (Samarov, 1993; Hristache et al., 2001a,b). Later on, the so-called Active Subspace (AS) method (Constantine et al., 2014; Zahm et al., 2020; Parente et al., 2020) proposed a dimension reduction method based on the minimization of an error bound of the mean squared error obtained via Poincaré inequalities. In the context of Bayesian inverse problems, similar methodologies have been proposed in (Cui et al., 2014; Zahm et al., 2022; Baptista et al., 2022; Li et al., 2023) in order to detect the parameter subspace which is the most informed by the data. Several extensions to nonlinear feature map have been recently proposed, including (Zhang et al., 2019a; Teng et al., 2021; Gruber et al., 2021; Bigoni et al., 2022; Romor et al., 2022). The common denominator of all those techniques is to build the feature map by aligning its Jacobian with the gradients of the model. Notably, (Bigoni et al., 2022) conducts a thorough analysis of this nonlinear dimension reduction problem using Poincaré Inequalities. This analysis, however, requires the feature map to have path-connected level sets, a condition which is hard to ensure in practice.

In this paper, we further explore different ways of learning nonlinear features using the model gradients. Specifically, we build the nonlinear features as the first components of a diffeomorphism defined on the input space, a solution originally considered in (Zhang et al., 2019a). Based on this, we pursue the analysis of (Bigoni et al., 2022) by applying Poincaré inequalities on either input space or feature space, *i.e.* the range of the diffeomorphism. That way, we obtain two different error bounds which we minimize in order to train the feature map. Let us note that such a strategy of optimizing an error bound is a well-tried strategy in many machine learning problems, see for instance the use of the evidence-based lower bound for optimizing variational auto-encoders (Kingma and Welling, 2013; Rezende et al., 2014). By considering such diffeomorphism-based feature maps, however, we drastically restrict the approximation class for the features. To circumvent this issue, we propose a dimension augmentation strategy to increase the expressiveness of the features while preserving the theoretical foundation of the method. The basic idea, originally proposed for neural ODEs (Dupont et al., 2019; Zhang et al., 2019b), is to introduce a new arbitrary random variable which is concatenated with the input random vector. Using this dimension augmentation trick results in a modified approximation class for the model, where the feature map is no longer a deterministic function but becomes a stochastic function. We note that another dimension augmentation trick has been used in (Romor et al., 2022) for gradient-based nonlinear feature learning, where the input variable is first embedded into an infinite Hilbert space in a non-bijective way before applying linear dimension reduction. Finally, we show that our method readily extends to vector-valued functions. In particular, by approximating the identity function over a given set of data, our methodology permits to

construct autoencoders (Nguyen et al., 2019) where the feature map is the encoder and the profile function is the decoder.

In practice, we construct the diffeomorphism, and consequently the feature map, using *coupling flow* based neural networks. This class of invertible networks is known for achieving favorable approximation properties (Teshima et al., 2020; Ishikawa et al., 2022; Lyu et al., 2022). Numerical comparisons with existing methods from the literature in (Bigoni et al., 2022; Zhang et al., 2019a; Teng et al., 2021) are presented. Our numerical demonstrations highlight the efficiency of our methodology in achieving accurate approximations across various high-dimensional test cases. Notably, in the small sample regime, employing dimension augmentation outperforms existing state-of-the-art methods.

The rest of the paper is organized as follow. In Section 2 we introduce the approximation problem and the proposed methodology. Section 3 explains how to use Poincaré Inequalities in order to derive error bounds which are then used to train the feature map. In Section 4 we introduce the dimension augmentation strategy. Section 5 generalizes the method to vector-valued functions and to autoencoders. In Section 6 we explain how to build the feature map using coupling flow networks. Finally, in Section 7, we compare numerically the proposed strategy with the existing nonlinear dimension reduction methods from the literature on several high dimensional functions, including a parametrized partial differential equation.

2 Problem statement and methodology

Let $u : \mathcal{X} \to \mathbb{R}$ be a function defined on the open set $\mathcal{X} \subseteq \mathbb{R}^d$ with $d \gg 1$. Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and $\mathbf{X} : (\Omega, \mathcal{A}) \to (\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ a measurable function, with $\mathcal{B}(\mathbb{R}^d)$ denoting the Borel σ-field on $\mathbb{R}^d$. We assume that $\mathbf{X}$ is absolutely continuous with Lebesgue on $\mathbb{R}^d$ and that its probability density function π is supported on $\mathcal{X}$. Our goal is to build a *feature map* $g : \mathcal{X} \to \mathbb{R}^m$ and a *profile function* $f : \mathbb{R}^m \to \mathbb{R}$ with *latent dimension* $m \leq d$ such that

$$\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}))^2] \leq \varepsilon, \tag{1}$$

for some prescribed tolerance ε, $\mathbb{E}[\cdot]$ denoting the expectation. Without additional assumptions, Problem (1) admits two trivial solutions which are $\{g = id; f = u; m = d\}$ and $\{g = u; f = id; m = 1\}$, where id denotes the identity function. These two solutions correspond to approximating u by g or by f directly without leveraging dimension reduction. Thus, Problem (1) makes sense only if we restrict g (or f) to a certain approximation class. In this paper, we propose to seek g in a set $\mathcal{G}_m$ defined as

$$\mathcal{G}_m = \left\{ \begin{array}{rccc} g: & \mathcal{X} & \to & \mathbb{R}^m \\ & x & \mapsto & (\varphi_1(x), \dots, \varphi_m(x)) \end{array} \;\middle|\; \varphi \in \mathcal{D} \right\}, \tag{2}$$

where $\mathcal{D}$ is a set of tractable $\mathcal{C}^1$-diffeomorphisms from $\mathcal{X}$ to $\mathbb{R}^d$, that is

$$\mathcal{D} \subseteq \left\{ \varphi \in \mathcal{C}^1(\mathcal{X}; \mathbb{R}^d) \text{ invertible with } \varphi^{-1} \in \mathcal{C}^1(\mathbb{R}^d; \mathcal{X}) \right\}.$$

Intuitively, the idea is to bend the input space $\mathcal{X}$ into a feature space $\varphi(\mathcal{X}) = \mathbb{R}^d$ where the relevant variables are the first m components of $\varphi(x) = (\varphi_1(x), \dots, \varphi_d(x))$. We demonstrate

in Section 3 that using such diffeomorphism-based feature maps $g \in \mathcal{G}_m$ ensures theoretical guarantees on the approximation error (1).

A naive strategy to solve (1) is to parametrize f and g, *e.g.*, with neural networks and to train f and g jointly by minimizing the $\mathbb{L}^2_\pi$ error $\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}))^2]$ estimated on a training set. If gradients of u are available, one can instead minimize the $\mathbb{H}^1_\pi$ error $\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}))^2] + \mathbb{E}[\|\nabla u(\mathbf{X}) - \nabla(f \circ g)(\mathbf{X})\|^2]$ estimated on a training set. The optimal latent dimension m is selected as the one which yields the smallest $\mathbb{L}^2_\pi$ error. In this paper we propose an alternative strategy which consists in two steps. In the first step, we train $g \in \mathcal{G}_m$ by minimizing an upper bound of the following $\mathbb{L}^2_\pi$ error

$$\mathcal{E}(g) = \min_{f:\mathbb{R}^m \to \mathbb{R}} \mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}))^2], \tag{3}$$

where the minimum is taken over the set of measurable functions without restrincting f to any approximation class. The quantity $\mathcal{E}(g)$ is called the reconstruction error. As detailed later in Section 3, this bound is derived using Poincaré inequalities and, in practice, it is estimated on a training set using gradient evaluations of u. In the second step, once g is built, we construct f, *e.g.*, as a neural network by solving a standard regression problem of $u(\mathbf{X})$ against $g(\mathbf{X})$, that is, we minimize the $\mathbb{L}^2_\pi$ error (or the $\mathbb{H}^1_\pi$ error) estimated on a training set. While minimizing an error bound for $\mathcal{E}(g)$ can yield, in principle, sub-optimal solutions compared to minimizing $\mathcal{E}(g)$ directly, we demonstrate that this strategy is computationally favorable. The main reason is that, while $\mathcal{E}(g)$ is not readily computable in practice (it requires minimizing over f), its upper bound turns out to be simple to estimate (it requires only evaluating ∇u).

By imposing $g \in \mathcal{G}_m$, we drastically restrict the approximation class for the feature map. In Section 4 we propose a dimension augmentation strategy to increase the expressiveness of the features while preserving the theoretical foundation of the method. The basic idea is to introduce a new arbitrary random variable $\boldsymbol{\Xi}$ taking values in $\Xi \subseteq \mathbb{R}^k$ and to consider the problem

$$\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}, \boldsymbol{\Xi}))^2] \leq \varepsilon, \tag{4}$$

instead of (1). By considering $g : \mathcal{X} \times \Xi \to \mathbb{R}^m$ to be the m first components of a diffeomorphism φ from $\mathcal{X} \times \Xi$ to $\mathbb{R}^{d+m}$, the above methodology directly applies. By doing this, we are no longer approximating $u(x)$ by a composition $f \circ g(x)$ but rather by a random variable $f \circ g(x, \boldsymbol{\Xi})$. Note that allowing the latent variable $g(x, \boldsymbol{\Xi})$ to be random is at the root of the variational autoencorders (Kingma et al., 2019). Eventually, a deterministic approximation to $u(x)$ can be obtained by taking the expectation over $\boldsymbol{\Xi}$ as follow

$$\widetilde{u}(x) = \mathbb{E}[f(g(x, \boldsymbol{\Xi}))], \tag{5}$$

and, by (4), the corresponding $\mathbb{L}^2_\pi$ error can still be bounded as $\mathbb{E}[(u(\mathbf{X}) - \widetilde{u}(\mathbf{X})))^2] \leq \varepsilon$. To emphasis the benefit of such dimension augmentation strategy, let us mention that for any $h \in \mathcal{C}^1(\mathcal{X}; \mathbb{R}^m)$, the feature map $g(x, \boldsymbol{\Xi}) = h(x) + \boldsymbol{\Xi}$ is the m first components of a diffeomorphism $\varphi : \mathcal{X} \times \Xi \to \mathbb{R}^{d+k}$ with $k = m$, see Equation (24) below. Thus, simply by adding a perturbation $\boldsymbol{\Xi}$ to the feature map, we replace the restrictive constraint $g \in \mathcal{G}_m$ in $u(x) \approx f \circ g(x)$ with the mild contraint $h \in \mathcal{C}^1(\mathcal{X}; \mathbb{R}^m)$ in $u(x) \approx \mathbb{E}[f(h(x) + \boldsymbol{\Xi})]$.

We end this section with an numerical illustration on a toy model which demonstrates the benefit of the proposed majorize-then-minimize strategy. For this experiment, we consider

$u(\mathbf{X}) = \cos(\|\mathbf{X}\|^2)$ for $\mathbf{X} \sim \mathcal{U}([0,1]^d)$ with $d = 20$ where $\|\cdot\|$ is the euclidean norm of $\mathbb{R}^d$. We observe on Figure 1 that the naive approach of jointly learning $f \circ g$ in $\mathbb{L}^2_\pi$ or $\mathbb{H}^1_\pi$ yields poor performance compared to our proposed two-step strategy, both when $g \in \mathcal{G}_m$ or $g \notin \mathcal{G}_m$. Additionally, the strategy of dimension augmentation contributes to further enhancing the effectiveness of our approach.

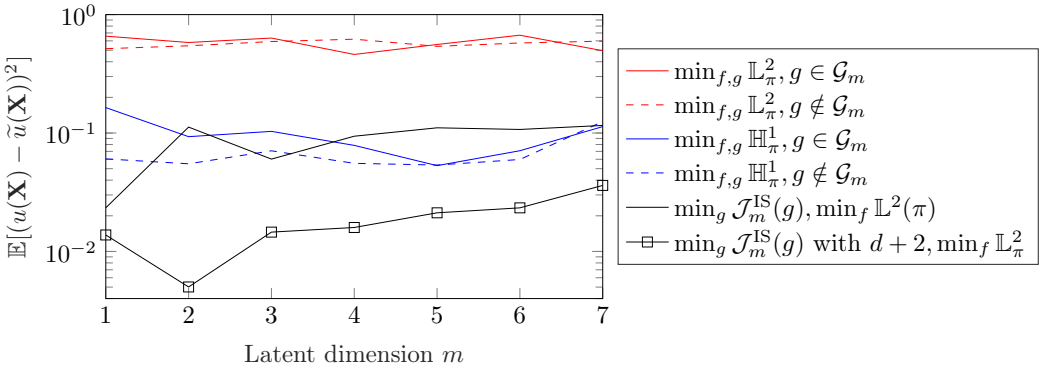


Figure 1: Toy benchmark $u(\mathbf{X}) = \cos(\|\mathbf{X}\|^2)$ for $\mathbf{X} \sim \mathcal{U}([0,1]^d)$ with $d = 20$. $\mathbb{L}^2_\pi$ error as a function of m of the approximation $\widetilde{u} = f \circ g$ or, when dimension augmentation is used, of $\widetilde{u}$ as in (5). Both f and g are trained as neural networks on a fixed training set of size 100 and the errors are estimated on a test set of size 10^4. Red and blue curves: we train f and g jointly by minimizing the $\mathbb{L}^2_\pi$ error (red) or the $\mathbb{H}^1_\pi$ error (blue), with either $g \notin \mathcal{G}_m$ (dashed) or $g \in \mathcal{G}_m$ (solid). Black curves: the proposed two-step strategy where we first train $g \in \mathcal{G}_m$ by minimizing the bound $\mathcal{J}^{\mathrm{IS}}_m(g)$ as in (9) with (squares) or without (solid) dimension augmentation, and then we minimize the $\mathbb{L}^2_\pi$ error over f. The function f is a fully connected neural network with 3 hidden layers of 50 neurons using the sigmoid activation function. The feature map $g \in \mathcal{G}_m$ is a block affine coupling flow with 4 layers, see Section 7. For $g \notin \mathcal{G}_m$, the feature map is a fully connected neural network with 3 hidden layers of 26 neurons for an overall number of parameters which is comparable to the above mentioned $g \in \mathcal{G}_m$ for fair comparison.

3 Upper bound of the approximation error using Poincaré Inequalities

In this section we derive upper bounds of the approximation error $\mathcal{E}(g)$. For any $g \in \mathcal{G}_m$, let us notice that the function f^*_g which realizes the minimium in (3) is the conditional expectation $f^*_g(z_m) = \mathbb{E}[u(\mathbf{X})|g(\mathbf{X}) = z_m]$. Denoting by $\varphi \in \mathcal{D}$ the diffeomorphism such that $g(x) = (\varphi_1(x), \ldots, \varphi_m(x))$, the change of variable $z = \varphi(x)$ permits to write

$$\mathcal{E}(g) = \mathbb{E}[(u(\mathbf{X}) - f^*_g \circ g(\mathbf{X}))^2],$$
$$\text{with} \quad f^*_g(z_m) = \int_{\mathbb{R}^{d-m}} u \circ \varphi^{-1}(z_m, z_\perp)\pi(z_\perp|z_m)\mathrm{d}z_\perp, \tag{6}$$

where $\pi(z_\perp|z_m)$ is the density of the conditional random vector $\mathbf{Z}_\perp|\mathbf{Z}_m = z_m$ with $(\mathbf{Z}_m, \mathbf{Z}_\perp) = \varphi(\mathbf{X})$. The notation $z = (z_m, z_\perp)$ refers to the coordinate splitting of the vector $z \in \mathbb{R}^d$ into its first m components $z_m \in \mathbb{R}^m$ and its last $d-m$ components $z_\perp \in \mathbb{R}^{d-m}$. Now, if we assume

that φ is such that the function $z_\perp \mapsto u \circ \varphi^{-1}(z_m, z_\perp)$ is constant for all z_m, then by (6) we have $f^*(z_m) = u \circ \varphi^{-1}(z_m, z_\perp)$ for all $z_\perp$. In particular with $z_\perp = (\varphi_{m+1}(x), \dots, \varphi_d(x))$ and $z_m = g(x)$, we deduce that

$$f_g^*(g(x)) = u \circ \varphi^{-1}(\varphi_1(x), \dots, \varphi_d(x)) = u(x), \tag{7}$$

for all $x \in \mathcal{X}$ and therefore $\mathcal{E}(g) = 0$. Next, we show how to build φ (and therefore g) in a way that the function $z_\perp \mapsto u \circ \varphi^{-1}(z_m, z_\perp)$ is as constant as possible. Our strategy is to minimize a certain norm of its gradient which we prove to be an upper bound for $\mathcal{E}(g)$. The following Poincaré Inequality will play a central role in our analysis.

Definition 1 (Poincaré Inequality on submanifold of $\mathbb{R}^d$) *For $\mathbf{Y}$ a continuous random vector taking values in a Riemannian submanifold $\mathcal{M} \subseteq \mathbb{R}^d$ equipped with the Euclidean metric of $\mathbb{R}^d$, the Poincaré constant $\mathbb{C}(\mathbf{Y}) \geq 0$ is defined as the smallest constant such that*

$$\mathbb{E}[(h(\mathbf{Y}) - \mathbb{E}[h(\mathbf{Y})])^2] \leq \mathbb{C}(\mathbf{Y})\mathbb{E}[\|\nabla h(\mathbf{Y})\|^2] \tag{8}$$

holds for any continuously differentiable function $h : \mathcal{M} \to \mathbb{R}$. Here, $\|\cdot\|$ denotes the Euclidean norm of $\mathbb{R}^d$ and $\nabla h(y) \in T_y(\mathcal{M})$ is the gradient of h at point $y \in \mathcal{M}$, where $T_y(\mathcal{M}) \subseteq \mathbb{R}^d$ is the tangent space of $\mathcal{M}$ at y. We say that $\mathbf{Y}$ satisfies Poincaré Inequality (8) if $\mathbb{C}(\mathbf{Y}) < +\infty$.

Finding sufficient conditions on $\mathbf{Y}$ which ensure $\mathbb{C}(\mathbf{Y}) < +\infty$ has been an active research field in functional analysis over the past decades. We refer to (Bakry et al., 2014, Chapter 4) for a discussion on the topic. In the next subsections, we propose two different upper bounds for $\mathcal{E}(g)$ which we derive using Poincaré Inequality (8) applied either on input space $\mathcal{X}$ or on feature space $\varphi(\mathcal{X})$, see Table 1 below for more details.

Input space (IS)		Feature space (FS)	
$\mathbf{Y}$	$= (\mathbf{X} \mid g(\mathbf{X}) = z_m)$	$\mathbf{Y}$	$= (\varphi(\mathbf{X}) \mid g(\mathbf{X}) = z_m)$
h	$= u_{\mid \mathcal{M}}$	h	$= u \circ (\varphi^{-1})_{\mid \mathcal{M}}$
$\mathcal{M}$	$= \{x \in \mathcal{X} \text{ s.t. } g(x) = z_m\}$	$\mathcal{M}$	$= \{(z_m, z_\perp) \in \mathbb{R}^d : z_\perp \in \mathbb{R}^{d-m}\}$
$T_{\mathbf{Y}}(\mathcal{M})$	$= \ker(\nabla g(\mathbf{X}))$	$T_{\mathbf{Y}}(\mathcal{M})$	$= \{(0, z_\perp) \in \mathbb{R}^d : z_\perp \in \mathbb{R}^{d-m}\}$
$\nabla h(\mathbf{Y})$	$= \Pi_{T_{\mathbf{Y}}(\mathcal{M})} \nabla u(\mathbf{Y})$	$\nabla h(\mathbf{Y})$	$= \Pi_{T_{\mathbf{Y}}(\mathcal{M})} \nabla (u \circ \varphi^{-1})(\mathbf{Y})$
	$= \Pi_{\ker(\nabla g(\mathbf{X}))} \nabla u(\mathbf{X})$		$= \nabla_\perp (u \circ \varphi^{-1})(\mathbf{Y})$

Table 1: Poincaré Inequality (8) applied either in input space or in feature space. Here, Π_V denotes the orthogonal projection on a subspace $V \subseteq \mathbb{R}^d$ and $\nabla_\perp = (\partial_{m+1}, \dots, \partial_d)$.

3.1 Poincaré Inequality in input space

As a warm-up exercice, assume there exists $g \in \mathcal{C}^1(\mathcal{X}; \mathbb{R}^m)$ and $f \in \mathcal{C}^1(\mathbb{R}^m; \mathbb{R})$ such that $u = f \circ g$. We denote by $\nabla g(x) \in \mathbb{R}^{m \times d}$ the Jacobian of g at x given by

$$\nabla g(x) = \begin{pmatrix} \partial_1 g_1(x) & \dots & \partial_d g_1(x) \\ \vdots & \ddots & \vdots \\ \partial_1 g_m(x) & \dots & \partial_d g_m(x) \end{pmatrix}.$$

By the chain rule, we have $\nabla u(x) = \nabla g(x)^\top \nabla f(g(x))$ and thus $\nabla u(x) \in \text{range}(\nabla g(x)^\top)$ for all $x \in \mathbb{R}^d$. The idea proposed in (Bigoni et al., 2022) is to enforce the Jacobian of g to be aligned with the gradient of u by minimizing

$$\mathcal{J}_m^{\text{IS}}(g) = \mathbb{E}[\|\nabla u(\mathbf{X}) - \Pi_{\text{range}(\nabla g(\mathbf{X})^\top)} \nabla u(\mathbf{X})\|^2], \tag{9}$$

where $\Pi_{\text{range}(\nabla g(\mathbf{X})^\top)}$ is the orthogonal projection onto $\text{range}(\nabla g(\mathbf{X})^\top)$. For more insight on this cost function, let us denote by $\mathcal{M}(\mathbf{X}) = \{x \in \mathcal{X} : g(x) = g(\mathbf{X})\}$ the (random) level set of g associated with the level $g(\mathbf{X})$ and let $u_{|\mathcal{M}(\mathbf{X})}$ be the restriction of u to $\mathcal{M}(\mathbf{X})$. Because the tangent space of $\mathcal{M}(\mathbf{X})$ at $\mathbf{X} \in \mathcal{M}(\mathbf{X})$ is the kernel of $\nabla g(\mathbf{X})$, see *e.g.* (Absil et al., 2008, Section 3.5.7), the gradient of $u_{|\mathcal{M}(\mathbf{X})}$ writes

$$\begin{aligned} \nabla u_{|\mathcal{M}(\mathbf{X})}(\mathbf{X}) &= \Pi_{\ker(\nabla g(\mathbf{X}))} \nabla u(\mathbf{X}) \\ &= \nabla u(\mathbf{X}) - \Pi_{\text{range}(\nabla g(\mathbf{X})^\top)} \nabla u(\mathbf{X}). \end{aligned} \tag{10}$$

Therefore, $\mathcal{J}_m^{\text{IS}}(g) = \mathbb{E}[\|\nabla u_{|\mathcal{M}(\mathbf{X})}(\mathbf{X})\|^2]$ so that minimizing $\mathcal{J}_m^{\text{IS}}(g)$ boils down to finding g which makes u as constant as possible on the level sets $\mathcal{M}(\mathbf{X})$ of g. The next proposition shows that the reconstruction error $\mathcal{E}(g)$ can be bounded with $\mathcal{J}_m^{\text{IS}}(g)$. The proof, given in Appendix A, is inspired from (Bigoni et al., 2022) which shows similar results for $g \notin \mathcal{G}_m$.

Proposition 2 *Let $u \in \mathcal{C}^1(\mathcal{X}; \mathbb{R})$ and $g \in \mathcal{G}_m$. Then, $\mathcal{J}_m^{\text{IS}}(g) = 0$ if and only if there exists $f \in \mathcal{C}^1(\mathbb{R}^m; \mathbb{R})$ such that $u = f \circ g$. Moreover, if*

$$\mathbb{C}^{\text{IS}}(\mathbf{X}|\mathcal{G}_m) := \sup_{g \in \mathcal{G}_m} \sup_{z_m \in \mathbb{R}^m} \mathbb{C}(\mathbf{X}|g(\mathbf{X}) = z_m) < +\infty,$$

then the reconstruction error $\mathcal{E}(g)$ as in (3) satisfies

$$\mathcal{E}(g) \leq \mathbb{C}^{\text{IS}}(\mathbf{X}|\mathcal{G}_m) \mathcal{J}_m^{\text{IS}}(g). \tag{11}$$

The above proposition suggests to build the feature map g as the solution to

$$\min_{g \in \mathcal{G}_m} \mathcal{J}_m^{\text{IS}}(g). \tag{12}$$

Such majorize-then-minimize strategy can be suboptimal if the bound (11) is loose, which can be the case if the constant $\mathbb{C}^{\text{IS}}(\mathbf{X}|\mathcal{G}_m)$ is large. Even though our numerical experiments reveal the good performances of this strategy, controlling $\mathbb{C}^{\text{IS}}(\mathbf{X}|\mathcal{G}_m)$ remains an open question which is left for future work. In the next section we derive another bound on $\mathcal{E}(g)$ which offers a viable path for controlling the associated Poincaré constant.

Remark 3 *The orthogonal projector $\Pi_{\text{range}(\nabla g(\mathbf{X})^\top)}$ writes*

$$\Pi_{\text{range}(\nabla g(\mathbf{X})^\top)} = \nabla g(\mathbf{X})^\top \left(\nabla g(\mathbf{X}) \nabla g(\mathbf{X})^\top \right)^{-1} \nabla g(\mathbf{X}).$$

Thus, using Pythagorean Theorem, the loss function $\mathcal{J}_m^{\text{IS}}(g)$ writes

$$\begin{aligned} \mathcal{J}_m^{\text{IS}}(g) &= \mathbb{E}[\|\nabla u(\mathbf{X})\|^2] - \mathbb{E}[\|\Pi_{\text{range}(\nabla g(\mathbf{X})^\top)} \nabla u(\mathbf{X})\|^2] \\ &= \mathbb{E}[\|\nabla u(\mathbf{X})\|^2] - \mathbb{E}\left[\nabla u(\mathbf{X})^\top \nabla g(\mathbf{X})^\top \left(\nabla g(\mathbf{X}) \nabla g(\mathbf{X})^\top \right)^{-1} \nabla g(\mathbf{X}) \nabla u(\mathbf{X}) \right]. \end{aligned} \tag{13}$$

The fact that $g \in \mathcal{G}_m$ ensure that $\nabla g(\mathbf{X})$ is almost surely full-rank, so that $\nabla g(\mathbf{X}) \nabla g(\mathbf{X})^\top$ is almost surely invertible.

3.2 Poincaré Inequality in feature space

As pointed out above in Equation (7), see also the proof of Proposition 2, it is sufficient that the function $z_\perp \mapsto (u \circ \varphi^{-1})(z_m, z_\perp)$ is constant in order to have $u = f \circ g$ for some $f : \mathbb{R}^d \to \mathbb{R}$. In this section, we propose to construct g by minimizing the $\mathbb{L}^2_\pi$ norm of the gradient of that function. With the change of variable $(z_m, z_\perp) = \varphi(x)$, this gradient writes

$$\begin{aligned}\nabla_\perp (u \circ \varphi^{-1})(z_m, z_\perp) &= \nabla_\perp \varphi^{-1}(z_m, z_\perp)^\top \nabla u(\varphi^{-1}(z_m, z_\perp)) \\ &= (\nabla \varphi^{-1}(\varphi(x))^\top \nabla u(x))_\perp \\ &= (\nabla \varphi(x)^{-\top} \nabla u(x))_\perp, \end{aligned} \tag{14}$$

where $(v)_\perp = (v_{m+1}, \dots, v_d) \in \mathbb{R}^{d-m}$ denotes the $d-m$ last components of a vector $v \in \mathbb{R}^d$. Thus we introduce

$$\mathcal{J}^{\mathrm{FS}}_m(\varphi) = \mathbb{E}[\|(\nabla \varphi(\mathbf{X})^{-\top} \nabla u(\mathbf{X}))_\perp\|^2] \tag{15}$$

as a cost function to train $\varphi \in \mathcal{D}$ and therefore $g = (\varphi_1, \dots, \varphi_m) \in \mathcal{G}_m$. The next proposition bounds $\mathcal{E}(g)$ with $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$. The proof is given in Appendix B

Proposition 4 *Let $u \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$, $\varphi \in \mathcal{D}$ and $m \leq d$. Then, $\mathcal{J}^{\mathrm{FS}}_m(\varphi) = 0$ if and only if there exists $f \in \mathcal{C}^1(\mathbb{R}^m, \mathbb{R})$ such that $u = f \circ g$, where $g = (\varphi_1, \dots, \varphi_m) \in \mathcal{G}_m$. Moreover, if*

$$\mathbb{C}^{\mathrm{FS}}_m(\mathbf{X}|\mathcal{D}) := \sup_{\varphi \in \mathcal{D}} \sup_{z_m \in \mathbb{R}^m} \mathbb{C}(\varphi(\mathbf{X})|g(\mathbf{X}) = z_m) < +\infty,$$

then the reconstruction error $\mathcal{E}(g)$ as in (3) satisfies

$$\mathcal{E}(g) \leq \mathbb{C}^{\mathrm{FS}}_m(\mathbf{X}|\mathcal{D}) \mathcal{J}^{\mathrm{FS}}_m(\varphi). \tag{16}$$

Inequality (16) suggests to build $g = (\varphi_1, \dots, \varphi_m)$ by minimizing $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$ over $\varphi \in \mathcal{D}$. However, unlike Problem (12), this variational problems might be ill-posed without imposing further constraints on φ. Indeed, notice that

$$\mathcal{J}^{\mathrm{FS}}_m(\alpha\varphi) = \frac{1}{\alpha^2} \mathcal{J}^{\mathrm{FS}}_m(\varphi), \tag{17}$$

holds for any $\varphi \in \mathcal{D}$ and $\alpha > 0$ so that, if $\mathcal{D}$ is a cone (that is $\alpha\varphi \in \mathcal{D}$ for $\varphi \in \mathcal{D}$ and $\alpha > 0$), then $\alpha \to \infty$ yields the trivial solution $\mathcal{J}^{\mathrm{FS}}_m(\alpha\varphi) \to 0$ for any $\varphi \in \mathcal{D}$. In fact, one can show that $\mathbb{C}^{\mathrm{FS}}_m(\mathbf{X}|\mathcal{D}) = +\infty$ whenever $\mathcal{D}$ is a cone, which makes the inequality (16) meaningless. To avoid this problematic situation, one option is to restrict $\mathcal{D}$ to contain only *normalizing flows* for $\mathbf{X}$ such that $\mathbf{Z} = \varphi(\mathbf{X}) \sim \mathcal{N}(0, I_d)$, where $\mathcal{N}(0, I_d)$ denotes the standard normal distribution on $\mathbb{R}^d$. This constraint is also convenient as it yields $\mathbb{C}(\varphi(\mathbf{X})|g(\mathbf{X}) = z_m) = 1$, see *e.g* (Bakry and Émery, 2006). Such a constraint is difficult to impose in practice. To address this issue we propose to penalize $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$ with the Kullback-Leibler divergence $\mathrm{D}_{\mathrm{KL}}(\varphi(\mathbf{X})||\mathbf{Z})$ from $\mathbf{Z} \sim \mathcal{N}(0, I_d)$ to $\varphi(\mathbf{X})$. Denoting by $\gamma(z) = (2\pi)^{-d/2} \exp(-\|z\|^2/2)$ the density of $\mathbf{Z}$ and by $\varphi^\sharp\gamma(x) = \gamma(\varphi(x))|\det \nabla\varphi(x)|$ the density of $\varphi^{-1}(\mathbf{Z})$, the Kullback-Leibler divergence reads

$$\begin{aligned}\mathrm{D}_{\mathrm{KL}}(\varphi(\mathbf{X})||\mathbf{Z}) &= \mathrm{D}_{\mathrm{KL}}(\mathbf{X}||\varphi^{-1}(\mathbf{Z})) \\ &= \int \log\left(\frac{\pi}{\varphi^\sharp\gamma}\right) \mathrm{d}\pi \\ &= \Omega + \int \frac{\|\varphi\|^2}{2} - \log|\det \nabla\varphi| \mathrm{d}\pi, \end{aligned}$$

where $\Omega = \int \log(\pi) \mathrm{d}\pi + d\log(2\pi)/2$ is a constant which is independent on φ. Finally we propose to build $g = (\varphi_1, \dots, \varphi_m)$ as the solution to:

$$\min_{\varphi \in \mathcal{D}} \mathcal{J}^{\mathrm{FS}}_{m,\lambda}(\varphi) := \mathbb{E}[\|(\nabla\varphi(\mathbf{X})^{-\top}\nabla u(\mathbf{X}))_\perp\|^2] + \lambda \mathbb{E}\left[\frac{\|\varphi(\mathbf{X})\|^2}{2} - \log|\det \nabla\varphi(\mathbf{X})|\right], \quad (18)$$

where $\lambda \geq 0$ is a penalization parameter to be tuned. While this penalization strategy do not permit to control the Poincaré constant $\mathbb{C}^{\mathrm{FS}}_m(\mathbf{X}|\mathcal{D})$, we show in Section 7 that it yields good performances on our numerical experiments.

Remark 5 *In (Romor et al., 2022), a non-bijective map $\varphi : \mathcal{X} \to \mathcal{H}$ is considered where $\mathcal{H}$ is an infinite-dimensional Hilbert space or a discretization of it $\mathcal{H} = \mathbb{R}^D, D \gg d$. Because $\nabla\varphi(\mathbf{X})$ is no longer invertible, the authors replace the inverse with the Moore–Penrose pseudoinverse in the loss function $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$. This map φ is parametrized as $\varphi(x) = R \circ \psi(x)$, where $\psi : \mathcal{X} \to \mathcal{H}$ is a given function (typically a feature map of a Reproducing Kernel Hilbert Space) and $R : \mathcal{H} \to \mathcal{H}$ is a rotation in $\mathcal{H}$ to be determined by minimizing $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$. Further investigation is required to gain a clearer understanding of whether this approach allows for bounding the reconstruction error $\mathcal{E}(g)$.*

3.3 Connection between $\mathcal{J}^{\mathrm{FS}}_m$, $\mathcal{J}^{\mathrm{IS}}_m$ and other methods

The next proposition shows that $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$ and $\mathcal{J}^{\mathrm{IS}}_m(g)$, albeit being derived with two different approaches, are closely related to each others. The proof is given in Appendix C.

Proposition 6 *Let $\varphi \in \mathcal{D}$ and $g = (\varphi_1, \dots, \varphi_m)$. For $M(\mathbf{X}) = \nabla\varphi(\mathbf{X})^{-1}\nabla\varphi(\mathbf{X})^{-\top}$, let $\|\cdot\|_{M(\mathbf{X})}$ be the norm on $\mathbb{R}^d$ associated with the scalar product $\langle v, w\rangle_{M(\mathbf{X})} = v^\top M(\mathbf{X}) w$. Then the cost function $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$ defined in (15) can be written as*

$$\mathcal{J}^{\mathrm{FS}}_m(\varphi) = \mathbb{E}\left[\left\|\nabla u(\mathbf{X}) - \Pi^{M(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)} \nabla u(\mathbf{X})\right\|^2_{M(\mathbf{X})}\right], \quad (19)$$

where $\Pi^{M(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}$ denotes the $\langle\cdot,\cdot\rangle_{M(\mathbf{X})}$-orthogonal projector on $\mathrm{range}(\nabla g(\mathbf{X})^\top)$. Furthermore, we have

$$\mathrm{Lip}(\varphi)^{-2}\mathcal{J}^{\mathrm{IS}}_m(g) \leq \mathcal{J}^{\mathrm{FS}}_m(\varphi) \leq \mathrm{Lip}(\varphi^{-1})^2\ \mathcal{J}^{\mathrm{IS}}_m(g), \quad (20)$$

where $\mathrm{Lip}(f) = \sup\{\frac{\|f(x)-f(y)\|}{\|x-y\|} : x \neq y\}$ denotes the Lipschitz constant of a function f.

It is also worth to mention that $\mathcal{J}^{\mathrm{FS}}_m(\varphi)$ is similar to the cost function used in (Zhang et al., 2019a) for the Nonlinear Level-set Learning (NLL). In that paper, the columns of $\nabla\varphi(\mathbf{X})^{-1} = [J_1(\mathbf{X}), \dots, J_d(\mathbf{X})]$ are normalized in L^2 norm which permits to remove the invariance (17). Specifically, they use the cost function

$$\mathcal{J}^{\mathrm{NLL}}_m(\varphi) = \mathbb{E}\left[\sum_{i=m+1}^{d}\left(\frac{J_i(\mathbf{X})^\top \nabla u(\mathbf{X})}{\|J_i(\mathbf{X})\|}\right)^2\right]. \quad (21)$$

According to (Teng et al., 2021), this method fails when u presents a critical point. To address this issue, the method DRiLLS (Teng et al., 2021) removes the invertibility constraint on φ

and penalizes the loss $\mathcal{J}_m^{\mathrm{NLL}}(\varphi)$ with a term which makes φ pseudo-invertible. Both NLL and DRiLLS require a large amount of training samples to accurately reduce the dimension. The following proposition shows that this cost function admits a similarly expression as $\mathcal{J}_m^{\mathrm{FS}}(\varphi)$ in (19) but with a slightly different norm for the gradient. The proof is given in Appendix D.

Proposition 7 *Let $\varphi \in \mathcal{D}$ and $g = (\varphi_1, \dots, \varphi_m)$. Then the cost function $\mathcal{J}_m^{\mathrm{NLL}}(\varphi)$ defined in (15) can be written as*

$$\mathcal{J}_m^{\mathrm{NLL}}(\varphi) = \mathbb{E}\left[\left\|\nabla u(\mathbf{X}) - \Pi^{D(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)} \nabla u(\mathbf{X})\right\|^2_{D(\mathbf{X})}\right], \tag{22}$$

where, for $J_i(\mathbf{X})$ being the i-th column of $\nabla\varphi(\mathbf{X})^{-1} = [J_1(\mathbf{X}), \dots, J_d(\mathbf{X})]$, the matrix $D(\mathbf{X})$ is defined as

$$D(\mathbf{X}) = \nabla\varphi(\mathbf{X})^{-1} \begin{pmatrix} \frac{1}{\|J_1(\mathbf{X})\|^2} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\|J_d(\mathbf{X})\|^2} \end{pmatrix} \nabla\varphi(\mathbf{X})^{-\top} = \sum_{i=1}^d \frac{J_i(\mathbf{X}) J_i(\mathbf{X})^\top}{\|J_i(\mathbf{X})\|^{-2}}.$$

Comparing the expressions (22), (19) and (9), we emphasis that the only difference between IF, FS and NLL is the norm used to measure the error between the gradient $\nabla u(\mathbf{X})$ and its best approximation in $\mathrm{range}(\nabla g(\mathbf{X})^\top)$ with respect to this norm.

We end this section by noticing that, in (Zhang et al., 2019a), a penalization term is employed in order to keep φ close to an isometry. The proposed penalized cost function is

$$\mathcal{J}_{m,\lambda}^{\mathrm{NLL}}(\varphi) = \mathcal{J}_m^{\mathrm{NLL}}(\varphi) + \lambda \mathbb{E}\left[(|\det \nabla\varphi(\mathbf{X})^{-1}| - 1)^2\right],$$

where the last term aims at keeping $|\det \nabla\varphi(\mathbf{X})^{-1}|$ close to one in the $\mathbb{L}^2_\pi$ sense. A similar penalization is used in (Teng et al., 2021).

Remark 8 *By letting $\varphi(x) = W^\top x$ be a linear isometry for some unitary matrix $W = [W_m, W_\perp] \in \mathbb{R}^{d\times d}$ with $W^\top W = I_d$, the feature map $g(x) = W_m^\top x$ is linear and*

$$\mathcal{J}_m^{\mathrm{IS}}(g) = \mathcal{J}_m^{\mathrm{FS}}(\varphi) = \mathcal{J}_m^{\mathrm{NLL}}(\varphi) = \mathbb{E}[\|\nabla u(\mathbf{X})\|^2] - \mathrm{trace}\left(W_m^\top H W_m\right),$$

where $H = \mathbb{E}[\nabla u(\mathbf{X}) \nabla u(\mathbf{X})^\top]$. Thus, minimizing $\mathcal{J}_m^{\mathrm{IS}}, \mathcal{J}_m^{\mathrm{FS}}$ or $\mathcal{J}_m^{\mathrm{NLL}}$ over W yields the feature map $g(x) = W_m^\top x$ where W_m contains the m-first eigenvectors of matrix H. This corresponds to the Active Subspace method (Constantine et al., 2014; Zahm et al., 2020).

4 Dimension augmentation to increase the expressiveness of the features

In this section we propose a dimension augmentation strategy which aims at increasing the approximation power of feature map g and, at the same time, preserving the theoretical foundation of the majorize then minimize strategy (see Section 3). While considering features $g \in \mathcal{G}_m$ such that $g(x) = (\varphi_1(x), \dots, \varphi_m(x))$ for some diffeomorphism $\varphi : \mathcal{X} \to \mathbb{R}^d$ is convenient theoretically (cf Propositions 2 and 4), it is rater restrictive in practice because not all functions $g : \mathcal{X} \to \mathbb{R}^m$ are the first components of a diffeomorphism $\varphi : \mathcal{X} \to \mathbb{R}^d$. However,

for any $g : \mathcal{X} \to \mathbb{R}^m$, there always exists a diffeomorphism $\varphi : \mathcal{X} \times \mathbb{R}^m \to \mathbb{R}^m \times \mathcal{X}$ such that $g(x) = (\varphi_1(x, 0), \dots, \varphi_m(x, 0))$, see Equation (24) below. In other words, augmenting the dimension permits to facilitate the construction of diffeomorphisms, as already noticed for neural ODEs (Dupont et al., 2019; Zhang et al., 2019b) or for normalizing flows in (Lyu et al., 2022; Huang et al., 2020).

The basic idea of dimension augmentation is to apply the methodology of the previous section to the dimension-augmented function $\overline{u} : \mathcal{X} \times \Xi \to \mathbb{R}$ defined by

$$\overline{u}(\mathbf{X}, \mathbf{\Xi}) = u(\mathbf{X}),$$

where $\Xi \subseteq \mathbb{R}^k$ for some $k \geq 1$ and where $\mathbf{\Xi}$ is a random vector independent on $\mathbf{X}$ with $\mathrm{supp}(\mathbf{\Xi}) = \Xi$. In our experiments, we chose $\mathbf{\Xi} \sim \mathcal{N}(0, I_m)$, independent of $\mathbf{X}$. For $m \leq d$, we consider the following set of feature maps

$$\mathcal{G}_m^k = \left\{ \begin{array}{rccc} g: & \mathcal{X} \times \Xi & \to & \mathbb{R}^m \\ & (x, \xi) & \mapsto & (\varphi_1(x, \xi), \dots, \varphi_m(x, \xi)) \end{array} \middle| \, \varphi \in \mathcal{D}^k \right\}$$
$$\mathcal{D}^k = \left\{ \varphi \in \mathcal{C}^1(\mathcal{X} \times \Xi; \mathbb{R}^{d+k}) \text{ invertible with } \varphi^{-1} \in \mathcal{C}^1(\mathbb{R}^{d+k}; \mathcal{X} \times \Xi) \right\}.$$

For $g \in \mathcal{G}_m^k$, a straightforward application of Propositions 2 and 4 permits to bound the reconstruction error

$$\mathcal{E}^k(g) = \min_{f:\mathbb{R}^m \to \mathbb{R}} \mathbb{E}[(\overline{u}(\mathbf{X}, \mathbf{\Xi}) - f \circ g(\mathbf{X}, \mathbf{\Xi}))^2], \tag{23}$$

by $\mathcal{E}^k(g) \leq \min\{\mathbb{C}^{\mathrm{IS}}(\mathbf{X}, \mathbf{\Xi} | \mathcal{G}_m^k) \mathcal{J}_{k,m}^{\mathrm{IS}}(g) \ ; \ \mathbb{C}_m^{\mathrm{FS}}(\mathbf{X}, \mathbf{\Xi} | \mathcal{D}^k) \mathcal{J}_{k,m}^{\mathrm{FS}}(\varphi)\}$, where

$$\mathcal{J}_{k,m}^{\mathrm{IS}}(g) = \mathbb{E}\left[\left\| \begin{pmatrix} \nabla u(\mathbf{X}) \\ 0_k \end{pmatrix} - \Pi_{\mathrm{range}(\nabla g(\mathbf{X}, \mathbf{\Xi})^\top)} \begin{pmatrix} \nabla u(\mathbf{X}) \\ 0_k \end{pmatrix} \right\|^2 \right],$$
$$\mathcal{J}_{k,m}^{\mathrm{FS}}(\varphi) = \mathbb{E}\left[\left\| \left(\nabla \varphi(\mathbf{X}, \mathbf{\Xi})^{-\top} \begin{pmatrix} \nabla u(\mathbf{X}) \\ 0_k \end{pmatrix} \right)_\perp \right\|^2 \right],$$

with the notation $0_k = (0, \dots, 0) \in \mathbb{R}^k$ and $(v)_\perp = (v_{m+1}, \dots, v_{d+k})$ for $v \in \mathbb{R}^{d+k}$. As before, the function f_g^* which realizes the minimium in (23) is the conditional expectation $f(z_m) = \mathbb{E}[u(\mathbf{X}) | g(\mathbf{X}, \mathbf{\Xi}) = z_m]$. However, by augmenting the dimension, we are no longer approximating $u(x)$ by a composition $f_g^* \circ g(x)$ as in the previous section, but rather by a random variable $f_g^* \circ g(x, \mathbf{\Xi})$. By taking the expectation over $\mathbf{\Xi}$, we obtain a deterministic approximation to $u(x)$ of the form

$$\widetilde{u}(x) = \mathbb{E}[f_g^* \circ g(x, \mathbf{\Xi})].$$

This structured approximation still leverages dimension reduction via the feature map $g : \mathcal{X} \times \Xi \to \mathbb{R}^m$. Because $\mathbf{X}$ and $\mathbf{\Xi}$ are independent, the reconstruction error decomposes as

$$\begin{aligned} \mathcal{E}^k(g) &= \mathbb{E}[(u(\mathbf{X}) - \widetilde{u}(\mathbf{X})) + (\widetilde{u}(\mathbf{X}) - f_g^* \circ g(\mathbf{X}, \mathbf{\Xi})))^2] \\ &= \mathbb{E}[(u(\mathbf{X}) - \widetilde{u}(\mathbf{X}))^2] + \mathbb{E}[(\widetilde{u}(\mathbf{X}) - f_g^* \circ g(\mathbf{X}, \mathbf{\Xi}))^2] \\ &= \mathbb{E}[(u(\mathbf{X}) - \widetilde{u}(\mathbf{X}))^2] + \mathbb{E}[\mathrm{Var}(f_g^* \circ g(\mathbf{X}, \mathbf{\Xi}) | \mathbf{X})], \end{aligned}$$

so that controlling $\mathcal{E}^k(g)$ permits to bound both the error $\mathbb{E}[(u(\mathbf{X}) - \widetilde{u}(\mathbf{X}))^2]$ and the expectation (over $\mathbf{X}$) of the point-wise variance $\mathrm{Var}(f_g^* \circ g(x, \mathbf{\Xi}))$. As reported in Section 7, our numerical experiments show that the variance $\mathbb{E}[\mathrm{Var}(f_g^* \circ g(\mathbf{X}, \mathbf{\Xi})|\mathbf{X})]$ is negligible compared to the error $\mathbb{E}[(u(\mathbf{X}) - \widetilde{u}(\mathbf{X}))^2]$. In the Global Sensitivity Analysis literature, the quantity $\mathbb{E}[\mathrm{Var}(f_g^* \circ g(\mathbf{X}, \mathbf{\Xi})|\mathbf{X})]$ is the *unnormalized total Sobol index* of $f_g^* \circ g$ associated to $\mathbf{\Xi}$, see *e.g* (Da Veiga et al., 2021). Thus if this term is negligible, it means that $f_g^* \circ g$ can be approximated by fixing $\mathbf{\Xi}$ to a nominal value.

We end this section by particularizing the above analysis to a specific diffeomorphism which reveals the advantage of the dimension augmentation. As already mentioned in Section 2, for any $h \in \mathcal{C}^1(\mathcal{X}; \mathbb{R}^m)$, the feature map $g(x, \xi) = h(x) + \xi$ corresponds to the m first components of the diffeomorphism $\varphi : \mathcal{X} \times \mathbb{R}^m \to \mathbb{R}^m \times \mathcal{X}$ defined by

$$\varphi(x, \xi) = \begin{pmatrix} h(x) + \xi \\ x \end{pmatrix} \quad \text{and} \quad \varphi^{-1}(z_1, z_2) = \begin{pmatrix} z_2 \\ z_1 - h(z_2) \end{pmatrix}. \tag{24}$$

The next proposition shows how to identify h via the cost functions $\mathcal{J}_{m,m}^{\mathrm{IS}}$ and $\mathcal{J}_{m,m}^{\mathrm{FS}}$. The proof is given in Appendix E.

Proposition 9 *For any $h \in \mathcal{C}^1(\mathcal{X}; \mathbb{R}^m)$ we let $g(x, \xi) = h(x) + \xi$ and φ as in (24). Then the loss function $\mathcal{J}_{m,m}^{\mathrm{FS}}(\varphi) = \mathbb{E}[\|\nabla u(\mathbf{X})\|^2]$ is a constant with respect to φ and therefore it does not permit to learn h. In addition we have*

$$\mathcal{J}_{m,m}^{\mathrm{IS}}(g) = \mathbb{E}[\|\nabla u(\mathbf{X})\|^2] - \mathbb{E}\left[u(\mathbf{X})^\top \nabla h(\mathbf{X})^\top (I_m + \nabla h(\mathbf{X}) \nabla h(\mathbf{X})^\top)^{-1} \nabla h(\mathbf{X}) \nabla u(\mathbf{X})\right].$$

We note that the difference with the non-augmented loss function $\mathcal{J}_m^{\mathrm{IS}}(g)$ in (13) is the presence of the identity matrix I_m which, conveniently, ensures $(I_m + \nabla h(\mathbf{X}) \nabla h(\mathbf{X})^\top)$ be to invertible.

5 Generalization to vector-valued functions and to autoencoders

In this section we present a generalization of the results of Section 3 to vector-valued functions. We then apply this generalization to unsupervised feature learning and autoencoders.

5.1 Case of vector-valued functions

We consider now a vector-valued function $u : \mathcal{X} \to \mathbb{R}^q$ with output dimension $q \geq 1$. As before, our goal is to build $g : \mathcal{X} \to \mathbb{R}^m$ and $f : \mathbb{R}^m \to \mathbb{R}^q$ such that

$$\mathbb{E}[\|u(\mathbf{X}) - f \circ g(\mathbf{X})\|^2] \leq \varepsilon,$$

for some prescribed tolerance threshold ε, where $\|\cdot\|$ denotes the Euclidean norm on $\mathbb{R}^q$. Following the methodology proposed in (Zahm et al., 2020), we first decompose the reconstruction error as follows

$$\mathcal{E}(g) = \min_{f : \mathbb{R}^m \to \mathbb{R}^q} \mathbb{E}[\|u(\mathbf{X}) - f \circ g(\mathbf{X})\|^2] = \sum_{i=1}^{q} \min_{f_i : \mathbb{R}^m \to \mathbb{R}} \mathbb{E}[(u_i(\mathbf{X}) - f_i \circ g(\mathbf{X}))^2]$$

and then we bound each term $\mathcal{E}_i(g) = \min_{f_i} \mathbb{E}[(u_i(\mathbf{X}) - f_i \circ g(\mathbf{X}))^2]$ using Poincaré Inequalities. Let us emphasis that $\nabla u(x) \in \mathbb{R}^{q \times d}$ denotes now the Jacobian of u and is given as

$$\nabla u(x) = \begin{pmatrix} \partial_1 u_1(x) & \dots & \partial_d u_1(x) \\ \vdots & \ddots & \vdots \\ \partial_1 u_q(x) & \dots & \partial_d u_q(x) \end{pmatrix}.$$

Using these notations, we generalize the definitions of $\mathcal{J}_m^{\mathrm{IS}}(g)$, $\mathcal{J}_m^{\mathrm{FS}}(\varphi)$ and $\mathcal{J}_{m,\lambda}^{\mathrm{FS}}(\varphi)$ as follows

$$\mathcal{J}_m^{\mathrm{IS}}(g) = \mathbb{E}[\|\nabla u(\mathbf{X})^\top - \Pi_{\mathrm{range}(\nabla g(\mathbf{X})^\top)} \nabla u(\mathbf{X})^\top\|_F^2] \tag{25}$$

$$\mathcal{J}_m^{\mathrm{FS}}(\varphi) = \mathbb{E}[\|U_\perp^\top \nabla \varphi(\mathbf{X})^{-\top} \nabla u(\mathbf{X})^\top\|_F^2] \tag{26}$$

$$\mathcal{J}_{m,\lambda}^{\mathrm{FS}}(\varphi) = \mathbb{E}[\|U_\perp^\top \nabla \varphi(\mathbf{X})^{-\top} \nabla u(\mathbf{X})^\top\|_F^2] + \lambda \mathbb{E}\left[\frac{\|\varphi(\mathbf{X})\|^2}{2} - \log|\det \nabla \varphi(\mathbf{X})|\right], \tag{27}$$

where $\|\cdot\|_F$ denotes the Frobenius norm such that $\|A\|_F^2 = \sum_{ij} A_{ij}^2$ and where $U_\perp \in \mathbb{R}^{(d+k)\times d}$ is the matrix given by

$$U_\perp = \begin{pmatrix} 0_{k\times d} \\ I_d \end{pmatrix}.$$

The following proposition, given without proof, generalizes Proposition 2 and Proposition 4 to the case of vector-valued functions.

Proposition 10 *Let $u \in \mathcal{C}^1(\mathcal{X}, \mathbb{R}^q)$, $\varphi \in \mathcal{D}$ and $g = (\varphi_1, \dots, \varphi_m)$ for some $m \leq d$. Then, $\mathcal{J}_m^{\mathrm{IS}}(g) = 0$ is equivalent to $\mathcal{J}_m^{\mathrm{FS}}(\varphi) = 0$ and also to having $u = f \circ g$ for some $f \in \mathcal{C}^1(\mathbb{R}^m, \mathbb{R}^q)$. In addition we have*

$$\mathcal{E}(g) \leq \mathbb{C}^{\mathrm{IS}}(\mathbf{X}|\mathcal{G}_m) \mathcal{J}_m^{\mathrm{IS}}(g),$$
$$\mathcal{E}(g) \leq \mathbb{C}_m^{\mathrm{FS}}(\mathbf{X}|\mathcal{D}) \mathcal{J}_m^{\mathrm{FS}}(\varphi),$$

where $\mathbb{C}^{\mathrm{IS}}(\mathbf{X}|\mathcal{G}_m)$ and $\mathbb{C}_m^{\mathrm{FS}}(\mathbf{X}|\mathcal{D})$ are as in Propositions 2 and 4.

5.2 Application to autoencoders

An interesting application of Proposition 10 is to consider $u : \mathcal{X} \to \mathbb{R}^d$ to be the identity function such that

$$u(\mathbf{X}) = \mathbf{X}. \tag{28}$$

This setup corresponds to the *unsupervised* learning task of estimating the random vector $\mathbf{X}$ itself given realizations $\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \dots$ of it. It is designed for scenarios in which $\mathbf{X}$ takes values in an unknown m-dimensional manifold $\mathcal{M} \subseteq \mathbb{R}^d$ which we want to learn. The feature map $g : \mathcal{X} \to \mathbb{R}^m$ can be interpreted as an *encoder* which extracts the *latent variable* $\mathbf{Z}_m = g(\mathbf{X})$. The profile function $f : \mathbb{R}^m \to \mathbb{R}^d$ corresponds to the *decoder* such that $f(\mathbf{Z}_m)$ yields an approximation to $\mathbf{X}$. Let us recall that, for any encoder $g \in \mathcal{G}_m$, the optimal decoder f_g^* writes as the conditional expectation $f_g^*(z_m) = \mathbb{E}[\mathbf{X}|g(\mathbf{X}) = z_m]$, or equivalently

$$f_g^*(z_m) = \mathbb{E}[\varphi^{-1}(z_m, \mathbf{Z}_\perp)|\mathbf{Z}_m = z_m],$$

where $(\mathbf{Z}_m, \mathbf{Z}_\perp) = \varphi(\mathbf{X})$. In other words, the optimal decoder admits a closed form expression involving only φ. Since $\nabla u(\mathbf{X}) = I_d$, the cost function $\mathcal{J}_m^{\mathrm{IS}}(g)$ defined in (25) becomes

$$\mathcal{J}_m^{\mathrm{IS}}(g) \overset{(28)}{=} \mathbb{E}[\|I_d - \Pi_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\|_F^2] = \mathbb{E}[d - \mathrm{trace}(\Pi_{\mathrm{range}(\nabla g(\mathbf{X}))})] = d - m,$$

for any $g \in \mathcal{G}_m$. Thus, $\mathcal{J}_m^{\mathrm{IS}}(g)$ is constant so that minimizing it does not permit to identify any relevant feature. On the other hand, the penalized cost function $\mathcal{J}_{m,\lambda}^{\mathrm{FS}}(\varphi)$ becomes

$$\mathcal{J}_{m,\lambda}^{\mathrm{FS}}(\varphi) \overset{(28)}{=} \mathbb{E}[\|U_\perp^\top \nabla\varphi(\mathbf{X})^{-\top}\|_F^2] + \lambda\mathbb{E}\left[\frac{\|\varphi(\mathbf{X})\|^2}{2} - \log|\det\nabla\varphi(\mathbf{X})|\right].$$

Let us mention that similar diffeomorphism-based autoencoders have been proposed in (Nguyen et al., 2019). In that paper, the encoder is also parametrized as $g = (\varphi_1, \ldots, \varphi_m)$, but the decoder is the sub-optimal decoder defined as $f(z_m) = \varphi^{-1}(z_m, 0_\perp)$, where $0_\perp = (0, \ldots, 0)$ is the zero vector in $\mathbb{R}^{d-m}$. The diffeomorphism φ is trained by minimizing directly the reconstruction error

$$\mathcal{L}_m(\varphi) = \mathbb{E}[\|\mathbf{X} - \varphi^{-1}(g(\mathbf{X}), 0_\perp)\|^2]. \tag{29}$$

Our numerical experiments show that this approach slightly outperforms our approach of minimizing the bound $\mathcal{J}_{m,\lambda}^{\mathrm{FS}}(\varphi)$. The reason for this might be that, on the considered test cases, the bound on the reconstruction error $\mathcal{E}(g) \leq \mathcal{L}_m(\varphi)$ is sharper compared to the bound we derive $\mathcal{E}(g) \leq \mathbb{C}_m^{\mathrm{FS}}(\mathbf{X}|\mathcal{D})\mathcal{J}_m^{\mathrm{FS}}(\varphi)$.

6 Invertible neural networks

We describe here a class of invertible neural networks employed for parameterizing and learning the diffeomorphism φ. Such neural networks find common applications in the field of *normalizing flows* within the unsupervised learning literature, as evidenced by works such as (Kobyzev et al., 2021; Dinh et al., 2015, 2017; Baptista et al., 2023), among others. The fundamental concept involves the composition of multiple *monotone triangular maps*, representing invertible functions where the k-th component depends solely on the first k variables and is monotone in the k-th variable. Moreover, this type of approximation tool is garnering increasing attention for estimating general diffeomorphisms, see (Teshima et al., 2020; Ishikawa et al., 2022; Lyu et al., 2022). In this section, we assume that the input domain $\mathcal{X}$ of u is

$$\mathcal{X} = \mathbb{R}^d,$$

and that input random vector $\mathbf{X}$ is fully supported on $\mathbb{R}^d$, meaning $\mathrm{supp}(\pi) = \mathbb{R}^d$. That way, we only consider diffeomorphisms φ from $\mathcal{X} = \mathbb{R}^d$ to $\mathbb{R}^d$.

Remark 11 *In practice, if $\mathcal{X} \subsetneq \mathbb{R}^d$, we can always consider diffeomorphism φ^0 that maps $\mathbb{R}^d$ to $\mathcal{X}$ and replace $u : \mathcal{X} \to \mathbb{R}$ with $u \circ \varphi^0 : \mathbb{R}^d \to \mathbb{R}$. In our numerical experiments for which $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_d$ is a product of intervals $\mathcal{X}_i = [a_i, b_i]$, we define $\varphi^0(x) = (\varphi_1^0(x_1), \ldots, \varphi_d^0(x_d))$ as a diagonal transformation such that $\varphi_i^0(\mathbf{X}_i) \sim \mathcal{N}(0,1)$ for all $1 \leq i \leq d$. This transform is obtained by letting*

$$\varphi_i^0(x_i) = F_{\mathcal{N}(0,1)}^{-1} \circ F_{\mathbf{X}_i}(x_i),$$

where $F_{\mathbf{X}_i}(t) = \int_{-\infty}^t \mathrm{d}\pi_i$ is the cumulative distribution function (CDF) of $\mathbf{X}_i \sim \pi_i$ and $F_{\mathcal{N}(0,1)}$ the CDF of the standard normal distribution $\mathcal{N}(0,1)$ on $\mathbb{R}$.

The term *affine coupling flows* refers to triangular functions whose k-th component is affine in the k-th variable, see (Papamakarios et al., 2021, Section 3.1). The following block affine coupling flows (BACF) are commonly encountered as they are easy to use and perform well in many applications.

Definition 12 (BACF) *Let $d \geq 2$. For two functions $s, t : \mathbb{R}^{\lfloor d/2 \rfloor} \to \mathbb{R}^{d-\lfloor d/2 \rfloor}$, the block affine coupling flow (BACF) $\Psi_{s,t} : \mathbb{R}^d \to \mathbb{R}^d$ is defined by*

$$\Psi_{s,t}(x) = \begin{pmatrix} x_{\leq \lfloor d/2 \rfloor} \\ x_{> \lfloor d/2 \rfloor} \odot \exp(s(x_{\leq \lfloor d/2 \rfloor})) + t(x_{\leq \lfloor d/2 \rfloor}) \end{pmatrix}, \tag{30}$$

where $\odot$ is the element-wise product and where $\exp(\cdot)$ is applied element-wise.

Notice that any function $\Psi_{s,t}$ as in (30) is invertible with

$$\Psi_{s,t}^{-1}(z) = \begin{pmatrix} z_{\leq \lfloor d/2 \rfloor} \\ (z_{> \lfloor d/2 \rfloor} - t(z_{\leq \lfloor d/2 \rfloor})) \odot \exp(-s(z_{\leq \lfloor d/2 \rfloor})) \end{pmatrix},$$

for any $z \in \mathbb{R}^d$, and the regularity of $\Psi_{s,t}$ and $\Psi_{s,t}^{-1}$ is exactly the one of the functions s, t. By composing multiple such functions, we obtain the following sets of diffeomorphisms.

Definition 13 (ℓ-layered BACF) *For $d, \ell \in \mathbb{N}^*$, we let*

$$\mathcal{D}_{\mathrm{BACF}}^{d,\ell} = \left\{ (P \circ \Psi_{s_\ell,t_\ell}) \circ \ldots \circ (P \circ \Psi_{s_1,t_1}) \;\middle|\; s_i, t_i \in \mathcal{C}^1(\mathbb{R}^{\lfloor \frac{d}{2} \rfloor}; \mathbb{R}^{d-\lfloor \frac{d}{2} \rfloor}) \right\}$$

where P is the block-coordinate permutation defined by $P(x) = (x_{> \lfloor \frac{d}{2} \rfloor}, x_{\leq \lfloor \frac{d}{2} \rfloor})$ for $x \in \mathbb{R}^d$.

In our experiments, we parametrize the functions s_i and t_i as shallow neural networks

$$s_i(x) = W_{s_i}^2 \sigma(W_{s_i}^1 x + b_{s_i}), \qquad t_i(x) = W_{t_i}^2 \sigma(W_{t_i}^1 x + b_{t_i}),$$

where $\sigma(t) = (1 + \exp(-t))^{-1}$ is the sigmoid function applied element-wise and where the matrices $W_{s_i}^1, W_{t_i}^1 \in \mathbb{R}^{\lfloor \frac{d}{2} \rfloor \times \lfloor \frac{d}{2} \rfloor}$ and $W_{s_i}^2, W_{t_i}^2 \in \mathbb{R}^{\lfloor \frac{d}{2} \rfloor \times (d - \lfloor \frac{d}{2} \rfloor)}$ and the vectors $b_{s_i}, b_{t_i} \in \mathbb{R}^{\lfloor \frac{d}{2} \rfloor}$ are to be determined. With this parametrization, any $\varphi \in \mathcal{D}_{\mathrm{BACF}}^{d,\ell}$ involves $\ell d(d+1)$ training parameters when d is even.

Remark 14 *Alternative choices have been proposed in the literature. For instance, (Lyu et al., 2022) takes the permutation P at random at each layer and uses single coordinate coupling flows (SACF) which only impacts the last coordinate of the input vector. (Zhang et al., 2019a) employs BACF with $s = 0$, yielding to a volume preserving diffeomorphism.*

7 Numerical Simulations

In this section, we train $\varphi \in \mathcal{D}_{\mathrm{BACF}}^{d,\ell}$ by minimizing either $\mathcal{J}_m^{\mathrm{IS}}(g)$ (`BACF_IS`(ℓ)) or $\mathcal{J}_{m,\lambda}^{\mathrm{FS}}(\varphi)$ (`BACF_FS`(ℓ)) estimated using a training set $\{(x^i, u(x^i), \nabla u(x^i))\}_{i=1}^{n_{\mathrm{train}}}$. For the dimension augmentation strategy of Section 4, we train $\varphi \in \mathcal{D}_{\mathrm{BACF}}^{d+k,\ell}$ by minimizing either $\mathcal{J}_{m,k}^{\mathrm{IS}}(g)$ (`BACF_IS`$_{d+k}(\ell)$) or $\mathcal{J}_{m,k,\lambda}^{\mathrm{FS}}(\varphi)$ (`BACF_FS`$_{d+k}(\ell)$) estimated using the same training set. Once

the feature map $g = (\varphi_1, \dots, \varphi_m)$ is built, we construct the profile function f as a fully connected neural network with 3 hidden layers of 50 neurons each and the sigmoid activation function. This network is trained by minimizing the empirical $\mathbb{L}^2_\pi$ error $\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}))^2]$ estimated again on the same training set. To evaluate the accuracy of the approximation, we compute the following errors on a randomly generated testing set of size n_{test}:

$$\text{MSE} = \mathbb{E}((u(\mathbf{X}) - f \circ g(\mathbf{X}))^2) \qquad \text{NRMSE} = \frac{\sqrt{\mathbb{E}((u(\mathbf{X}) - f \circ g(\mathbf{X}))^2)}}{\max\limits_{1 \le i \le n_{\text{test}}} u(x^i) - \min\limits_{1 \le i \le n_{\text{test}}} u(x^i)}$$

$$\text{RL}_2 = \sqrt{\frac{\mathbb{E}((u(\mathbf{X}) - f \circ g(\mathbf{X}))^2)}{\mathbb{E}(u(\mathbf{X})^2)}} \qquad \text{RL}_1 = \frac{\mathbb{E}(|u(\mathbf{X}) - f \circ g(\mathbf{X})|)}{\mathbb{E}(|u(\mathbf{X})|)}.$$

The column #Param is the total number of parameters used for g. All the approximation errors are written $\mu \pm \sigma$ where μ and σ are the mean and the standard deviation over 15 runs of the algorithm.

All the simulations have been implemented with PyTorch 2.01 and tested on a laptop with a 4.7 GHz Intel Core i7 CPU and 32GB of DRAM memory. The cost functions are minimized with the ADAM optimizer, a multi-start procedure is applied with 8 different randomly chosen starting points, after 300 ADAM steps the best candidate is selected and then optimized for 3000 additional steps. All this procedure is done with a learning rate of 0.01. f is also trained with the ADAM optimizer but without multi-start. The same learning rate is used and the learning process is stopped after 5000 ADAM steps or when the loss function reduces to 10^{-8}.

7.1 High dimensional function approximation

In this section we compare `BACF_IS`(ℓ), `BACF_FS`(ℓ), `BACF_IS`$_{d+k}(\ell)$ and `BACF_FS`$_{d+k}(\ell)$ with the existing solutions for nonlinear dimension reduction: `NLL` (Zhang et al., 2019a), `DRiLLS` (Teng et al., 2021) and with the Polynomial Feature Map (`PFM`, see (Bigoni et al., 2022)). All those algorithms use gradient information to perform nonlinear dimension reduction on small training sets. We consider the following benchmark functions

$$u_1(x) = \cos\left(x_d \exp\left(\sum_{i=1}^{d-1} \sigma(x_i)\right)\right), \qquad u_2(x) = \sin(\|\mathbf{x}\|^2),$$

$$u_3(x) = \exp\left(\frac{1}{d}\sum_{i=1}^{d} \sin(x_i) e^{\cos(x_i)}\right),$$

where σ is the sigmoid function. These functions are defined on $\mathcal{X} = [-1, 1]^d$ for u_1, $\mathcal{X} = [0, 1]^d$ or $\mathcal{X} = [-1, 1]^d$ for u_2, $\mathcal{X} = [-\frac{\pi}{2}, \frac{\pi}{2}]^d$ for u_3, and $\mathbf{X}$ is uniformly distributed on $\mathcal{X}$. The training points $\{x^i\}_{i=1}^{n_{\text{train}}}$ are sampled through a latin hypercube sampling optimized with maximin criterion, using the scikit-optimize library (Head et al., 2022). The testing set is made of $n_{\text{test}} = 10^4$ independent copies of $\mathbf{X}$.

We choose the above benchmark functions to perform a fair comparison of all dimension reduction methods. u_1 is a composition of cosine with the first component of a single-coordinate affine coupling flows, hence it is well suited for our approximation method (BACF).

u_1 on $\mathcal{X} = [-1,1]^8$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 50$	BACF_IS(2)	144	$(\mathbf{3.45} \pm 2.45) \times \mathbf{10^{-5}}$	$\mathbf{1.15} \pm 0.41$	$\mathbf{0.65} \pm 0.23$	$\mathbf{0.52} \pm 0.23$
	$\text{BACF\_IS}_{d+2}(2)$	220	$(6.18 \pm 7.90) \times 10^{-5}$	1.41 ± 0.82	0.80 ± 0.47	0.67 ± 0.48
	BACF_FS(2)	144	$(1.16 \pm 2.14) \times 10^{-4}$	1.82 ± 1.24	1.05 ± 0.71	0.87 ± 0.71
	$\text{BACF\_FS}_{d+2}(2)$	220	$(2.31 \pm 5.86) \times 10^{-4}$	2.11 ± 2.32	1.20 ± 1.32	1.06 ± 1.37
	PFM	-	$(1.91 \pm 0.00) \times 10^{-2}$	30.01 ± 0.00	16.18 ± 0.00	14.16 ± 0.02
u_1 on $\mathcal{X} = [-1,1]^{12}$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 50$	BACF_IS(2)	312	$(4.33 \pm 7.33) \times 10^{-5}$	1.09 ± 0.93	0.59 ± 0.50	0.49 ± 0.42
	$\text{BACF\_IS}_{d+2}(2)$	420	$(3.56 \pm 1.41) \times 10^{-5}$	1.27 ± 0.27	0.69 ± 0.15	0.56 ± 0.12
	BACF_FS(2)	312	$(\mathbf{2.65} \pm 3.18) \times \mathbf{10^{-5}}$	$\mathbf{0.96} \pm 0.57$	$\mathbf{0.52} \pm 0.31$	$\mathbf{0.43} \pm 0.24$
	$\text{BACF\_FS}_{d+2}(2)$	420	$(4.23 \pm 5.66) \times 10^{-5}$	1.15 ± 0.83	0.62 ± 0.44	0.51 ± 0.37
	PFM	-	$(2.00 \pm 0.00) \times 10^{-2}$	29.00 ± 0.00	16.59 ± 0.00	14.56 ± 0.02

Table 2: Approximation errors for u_1 with $\mathcal{X} = [-1,1]^d$, $n_{\text{train}} = 50$ and $m = 1$, $\lambda = 10^{-4}$

u_2 is well suited for PFM since $u = f \circ g$ for the polynomial $g(x) = \|x\|^2 = x_1^2 + \ldots + x_d^2$. Finally u_3 is a composition of the exponential function with a complex non polynomial function. As u_3 cannot be analytically decomposed with a polynomial feature map nor a coupling flows, neither PFM nor BACF are favored.

Table 2, Table 3 and Table 4 present the results for u_1, u_2 and u_3, respectively. Table 2 confirms that BACF_IS and BACF_FS are able to recover u_1 accurately with only 50 training points, while PFM encounters difficulties with such limited training set. In turns, Table 3 confirms that PFM outperforms the other methods on u_2. However, on $\mathcal{X} = [0,1]^{20}$, BACF_IS_{d+2} achieves better results with 100 training points compared to PFM with 100 training points and compared to NLL and DRiLLS with 500 training points. Also, on $\mathcal{X} = [-1,1]^8$ and $\mathcal{X} = [-1,1]^{12}$, BACF_IS_{d+2} achieves better results with only 500 training points compared to DRiLLS with 2500 training points. We do not compare with NLL on $\mathcal{X} = [-1,1]^8$ and $\mathcal{X} = [-1,1]^{12}$ because, as noticed in (Zhang et al., 2019a; Teng et al., 2021), NLL is not appropriate for functions with critical point in the interior of its domain.

Overall, we observe that BACF_IS with dimension augmentation performs generally better than BACF_IS without dimension augmentation and than BACF_FS for a comparable number of training parameters. Although dimension augmentation leads to a significant improvement in the performance of BACF_IS, it appears not to have a notable effect on BACF_FS. It is worth to note that, on Table 4, BACF_IS with dimension augmentation outperforms all the other methods on this benchmark, especially PFM algorithm which cannot achieve relative errors under 10%. These results confirm that BACF_IS with dimension augmentation is well suited for approximating high dimensional function.

Finally, Table 5 reports the approximation error $\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}, \mathbf{\Xi}))^2]$ and the maximum of the pointwise variance $\max_x \text{Var}(f \circ g(x, \mathbf{\Xi}))$ over the test set when dimension augmentation is employed. As already mentioned in Section 4, we observe that $\mathbb{E}[\text{Var}(f \circ g(x, \mathbf{\Xi}))]$ is negligible compared to $\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}, \mathbf{\Xi}))^2]$.

u_2 on $\mathcal{X} = [0,1]^{20}$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 100$	BACF_IS(5)	2100	$(3.83 \pm 3.85) \times 10^{-2}$	9.11 ± 3.59	26.14 ± 10.30	18.98 ± 6.00
	BACF_IS$_{d+2}$(4)	2024	$(\mathbf{8.01} \pm 4.88) \times \mathbf{10^{-3}}$	$\mathbf{4.29} \pm 1.27$	$\mathbf{12.32} \pm 3.65$	$\mathbf{8.57} \pm 2.35$
	BACF_FS(5)	2100	$(7.80 \pm 1.67) \times 10^{-2}$	13.89 ± 1.43	39.99 ± 4.12	31.51 ± 2.98
	BACF_FS$_{d+2}$(4)	2024	$(5.84 \pm 2.59) \times 10^{-2}$	11.81 ± 2.56	33.73 ± 7.32	26.08 ± 6.13
	PFM	230	$(9.73 \pm 4.36) \times 10^{-2}$	15.23 ± 3.47	43.77 ± 9.98	13.38 ± 7.86
$n_{\text{train}} = 500$	BACF_IS(5)	2100	$(1.42 \pm 0.95) \times 10^{-3}$	1.81 ± 0.53	5.22 ± 1.53	3.34 ± 1.36
	BACF_IS$_{d+2}$(4)	2024	$(1.91 \pm 2.00) \times 10^{-3}$	1.96 ± 0.96	5.64 ± 2.77	4.21 ± 3.09
	BACF_FS(5)	2100	$(1.70 \pm 0.82) \times 10^{-3}$	2.01 ± 0.46	5.75 ± 1.32	3.99 ± 1.12
	BACF_FS$_{d+2}$(4)	2100	$(1.92 \pm 1.19) \times 10^{-3}$	2.12 ± 0.57	6.07 ± 1.62	4.32 ± 1.46
	PFM	230	$(\mathbf{1.49} \pm 4.13) \times \mathbf{10^{-4}}$	$\mathbf{0.39} \pm 0.48$	$\mathbf{1.13} \pm 1.39$	$\mathbf{0.44} \pm 0.15$
	NLL	-	-	5.09	-	7.46
	DRiLLS	-	-	28.73	-	79.29

u_2 on $\mathcal{X} = [-1,1]^8$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 500$	BACF_IS(6)	432	$(9.50 \pm 31.20) \times 10^{-2}$	8.05 ± 13.15	23.86 ± 38.98	21.23 ± 39.58
	BACF_IS$_{d+2}$(4)	440	$(1.00 \pm 3.44 \times 10^{-2}$	2.47 ± 4.35	7.37 ± 12.93	5.75 ± 12.07
	BACF_FS(6)	432	$(2.94 \pm 1.50) \times 10^{-2}$	8.37 ± 1.93	25.00 ± 5.76	17.36 ± 5.09
	BACF_FS$_{d+2}$(4)	440	$(4.69 \pm 3.30) \times 10^{-2}$	9.94 ± 4.28	29.49 ± 12.68	23.17 ± 11.38
	PFM	44	$(\mathbf{2.33} \pm 6.52) \times \mathbf{10^{-8}}$	$(\mathbf{4.16} \pm 6.62) \times \mathbf{10^{-3}}$	$(\mathbf{1.24} \pm 1.98) \times \mathbf{10^{-2}}$	$(\mathbf{1.27} \pm 0.35) \times \mathbf{10^{-3}}$
$n_{\text{train}} = 2500$	DRiLLS	-	-	9.18	-	15.90

u_2 on $\mathcal{X} = [-1,1]^{12}$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 500$	BACF_IS(7)	1092	$(5.94 \pm 0.94) \times 10^{-3}$	3.84 ± 0.28	11.03 ± 0.82	7.93 ± 0.46
	BACF_IS$_{d+4}$(4)	1088	$(2.88 \pm 10.47) \times 10^{-2}$	3.43 ± 7.76	9.78 ± 22.10	7.53 ± 17.38
	BACF_FS(7)	1092	$(3.57 \pm 4.74) \times 10^{-2}$	8.32 ± 4.48	23.82 ± 12.82	16.20 ± 10.35
	BACF_FS$_{d+4}$(4)	1088	$(7.69 \pm 13.94) \times 10^{-2}$	9.60 ± 10.00	27.24 ± 28.35	20.27 ± 22.21
	PFM	90	$(\mathbf{3.00} \pm 8.92) \times \mathbf{10^{-8}}$	$(\mathbf{5.01} \pm 7.32) \times \mathbf{10^{-4}}$	$(\mathbf{1.43} \pm 2.09) \times \mathbf{10^{-3}}$	$(\mathbf{1.45} \pm 5.32) \times \mathbf{10^{-5}}$
$n_{\text{train}} = 2500$	DRiLLS	-	-	26.86	-	74.17

Table 3: Approximation errors for u_2 with $m = 1$, the NLL and DRiLLS results are copied from (Teng et al., 2021) Table 3, $\lambda = 10^{-4}$

u_3 on $\mathcal{X} = [-\frac{\pi}{2}, \frac{\pi}{2}]^8$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 100$	BACF_IS(6)	432	$(3.06 \pm 1.74) \times 10^{-2}$	4.75 ± 1.25	14.32 ± 3.77	10.63 ± 2.72
	$\text{BACF\_IS}_{d+2}(4)$	440	$(\mathbf{2.61} \pm 2.45) \times \mathbf{10^{-3}}$	$\mathbf{1.40} \pm 0.64$	$\mathbf{3.98} \pm 1.82$	$\mathbf{2.54} \pm 1.01$
	BACF_FS(6)	432	$(2.06 \pm 0.58) \times 10^{-1}$	12.40 ± 1.83	37.94 ± 5.60	29.41 ± 4.41
	$\text{BACF\_FS}_{d+2}(4)$	440	$(1.57 \pm 0.77) \times 10^{-1}$	10.80 ± 2.39	33.12 ± 7.33	23.98 ± 3.25
	PFM	44	$(1.93 \pm 0.24) \times 10^{-1}$	12.03 ± 0.79	37.13 ± 2.43	30.78 ± 2.22
$n_{\text{train}} = 500$	BACF_IS(6)	432	$(5.15 \pm 2.14) \times 10^{-3}$	2.01 ± 0.34	6.00 ± 1.02	4.46 ± 0.99
	$\text{BACF\_IS}_{d+2}(4)$	440	$(\mathbf{4.50} \pm 3.08) \times \mathbf{10^{-4}}$	$\mathbf{0.56} \pm 0.22$	$\mathbf{1.67} \pm 0.65$	$\mathbf{1.04} \pm 0.44$
	BACF_FS(6)	432	$(9.22 \pm 2.79) \times 10^{-3}$	2.77 ± 0.42	8.12 ± 1.22	5.61 ± 0.85
	$\text{BACF\_FS}_{d+2}(4)$	440	$(7.47 \pm 4.78) \times 10^{-3}$	2.16 ± 0.71	6.96 ± 2.30	4.81 ± 1.45
	PFM	44	$(1.94 \pm 0.17) \times 10^{-1}$	12.08 ± 0.56	37.69 ± 1.71	31.26 ± 1.44
u_3 on $\mathcal{X} = [-\frac{\pi}{2}, \frac{\pi}{2}]^{12}$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 100$	BACF_IS(7)	1092	$(2.03 \pm 1.02) \times 10^{-1}$	6.48 ± 1.54	34.25 ± 8.12	24.80 ± 5.56
	$\text{BACF\_IS}_{d+4}(4)$	1088	$(\mathbf{4.56} \pm 5.82) \times \mathbf{10^{-2}}$	$\mathbf{3.26} \pm 1.83$	$\mathbf{14.51} \pm 8.15$	$\mathbf{8.66} \pm 4.29$
	BACF_FS(7)	1092	$(3.19 \pm 1.36) \times 10^{-1}$	10.59 ± 2.08	43.50 ± 8.56	32.17 ± 4.24
	$\text{BACF\_FS}_{d+4}(4)$	1088	$(3.57 \pm 1.82) \times 10^{-1}$	10.77 ± 2.51	45.54 ± 10.57	32.25 ± 4.74
	PFM	90	$(7.08 \pm 3.43) \times 10^{-2}$	9.37 ± 1.98	23.32 ± 4.92	17.14 ± 1.97
$n_{\text{train}} = 500$	BACF_IS(7)	1092	$(1.67 \pm 1.08) \times 10^{-2}$	1.63 ± 0.52	9.54 ± 3.06	7.27 ± 2.87
	$\text{BACF\_IS}_{d+4}(4)$	1088	$(\mathbf{1.70} \pm 0.87) \times \mathbf{10^{-4}}$	$\mathbf{0.66} \pm 0.18$	$\mathbf{3.06} \pm 0.81$	$\mathbf{2.10} \pm 0.38$
	BACF_FS(7)	1092	$(9.38 \pm 1.93) \times 10^{-3}$	1.55 ± 0.15	7.61 ± 0.76	5.14 ± 0.40
	$\text{BACF\_FS}_{d+4}(4)$	1088	$(4.28 \pm 1.78) \times 10^{-3}$	1.28 ± 0.26	5.02 ± 1.01	3.56 ± 0.78
	PFM	90	$(1.24 \pm 0.07) \times 10^{-1}$	12.66 ± 0.38	31.52 ± 0.95	25.85 ± 0.53

Table 4: Approximation errors for u_3 on $\mathcal{X} = [-\frac{\pi}{2}, \frac{\pi}{2}]^d$ and with $m = 1$, $\lambda = 10^{-4}$

u_2 on $\mathcal{X} = [0,1]^{20}$			
Points	Structure	$\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}, \mathbf{\Xi}))^2]$	$\max_x \text{Var}(f \circ g(x, \mathbf{\Xi}))$
$n_{\text{train}} = 100$	$\text{BACF\_IS}_{d+2}(4)$	8.01×10^{-3}	4.22×10^{-9}
	$\text{BACF\_FS}_{d+2}(4)$	5.84×10^{-2}	9.81×10^{-3}
$n_{\text{train}} = 500$	$\text{BACF\_IS}_{d+2}(4)$	1.91×10^{-3}	1.43×10^{-10}
	$\text{BACF\_FS}_{d+2}(4)$	1.92×10^{-3}	1.71×10^{-5}
u_2 on $\mathcal{X} = [-1,1]^8$			
Points	Structure	$\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}, \mathbf{\Xi}))^2]$	$\max_x \text{Var}(f \circ g(x, \mathbf{\Xi}))$
$n_{\text{train}} = 500$	$\text{BACF\_IS}_{d+2}(4)$	1.00×10^{-2}	2.07×10^{-10}
	$\text{BACF\_FS}_{d+2}(4)$	4.69×10^{-2}	5.79×10^{-6}
u_2 on $\mathcal{X} = [-1,1]^{12}$			
Points	Structure	$\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}, \mathbf{\Xi}))^2]$	$\max_x \text{Var}(f \circ g(x, \mathbf{\Xi}))$
$n_{\text{train}} = 500$	$\text{BACF\_IS}_{d+4}(4)$	2.88×10^{-2}	1.90×10^{-10}
	$\text{BACF\_FS}_{d+4}(4)$	7.69×10^{-2}	2.52×10^{-5}

Table 5: Approximation error $\mathbb{E}[(u(\mathbf{X}) - f \circ g(\mathbf{X}, \mathbf{\Xi}))^2]$ and maximum of the pointwise variance $\max_x \text{Var}(f \circ g(x, \mathbf{\Xi}))$ estimated on a test set of size 10^4 where $\text{Var}(f \circ g(x, \mathbf{\Xi}))$ is estimated using 100 samples of $\mathbf{\Xi}$. All these numbers are averaged over 15 realizations of the training set.

7.2 The thermal block problem

In this section we test our method on the thermal bloc problem, where the function to approximate is defined via the solution of a partial differential equation (PDE) which we solve with the finite element method (Ern and Guermond, 2004). The gradient is computed using adjoint-state method (Plessix, 2006). We use here the same settings as in (Teng et al., 2021) and we consider the diffusion equation

$$-\nabla_s \cdot (\kappa \nabla_s v) = 0 \quad \text{in } \Omega = [0,1]^2, \tag{31}$$

where $s = (s_1, s_2) \in \Omega$ are the spatial coordinates, and ∇_s refers to the gradient with respect to s. Homogeneous Dirichlet boundary conditions are imposed on the upper boundary of the domain $\partial\Omega_D$, and homogeneous Neumann boundary condition is imposed on the left and right edges $\partial\Omega_{N,0}$. We impose a unit flux on the lower boundary $\partial\Omega_{N,1}$. The domain is uniformly partitioned into d sub-domains $\{\Omega_i\}_{i=1}^d$ and the diffusion coefficient κ is piecewise constant such that $\kappa(s,x) = \sum_{i=1}^d x_i \mathbf{1}_{\Omega_i}(s)$ with $x \in \mathcal{X} = [0.1, 10]^d$. For $\mathbf{X} \sim \mathcal{U}(\mathcal{X})$, we define $u(\mathbf{X})$ as

$$u(\mathbf{X}) = \int_{\partial\Omega_{N,1}} v(s, \mathbf{X}) \mathrm{d}s.$$

All algorithms are trained on 15 different training sets of size n_{train} and then tested on a testing set of size $n_{\text{test}} = 10^4$.

The results are presented in Table 6 and Table 7. Again we notice that `BACF_IS` with dimension augmentation is clearly the best method we tried. When $d = 16$ and with 500 training points, it outperforms `NLL` and `DRiLLS` with 2500 training points and `PFM` with 500 training points. Table 7 shows that `BACF_IS` with dimension augmentation also performs very well for larger dimension $d = 36$, achieving a NRMSE of 4% and a RL_1 of 5% with only 200 training points. When $d = 36$ we notice that the dimension augmentation strategy does not provide as much improvements as for $d = 16$. The scatter plot $\{(g(x^i), u(x^i))\}_{i \geq 1}$ from Figure 2 also shows that `BACF_IS` can successfully reduce the input dimension for $d = 16$ and $d = 36$ with very little training sets.

7.3 Autoencoders

We compare now the two strategies proposed in Section 5.2 to train autoencoders. Here the goal is to build the best encoder g and decoder f that minimize the empirical reconstruction loss $\frac{1}{n}\sum_{i=1}^{n_{test}} \|x^{(i)} - f \circ g(x^{(i)})\|^2$ on a testing set $\{x^{(i)}\}_{i=1}^{n_{\text{test}}}$. We propose to compare the two strategies presented in Section 5.2 on a synthetic dataset in $\mathbb{R}^d$ where the points belong to a submanifold of intrinsic dimension $m_{\text{int}} < d$. We generate this submanifold with the following application:

$$\begin{array}{rccc} \Psi_Q : & \mathbb{R}^{m_{\text{int}}} & \to & \mathbb{R}^d \\ & Z & \mapsto & (Z^\top Q_1 Z, \ldots, Z^\top Q_d Z) \end{array}$$

where $Q = (Q_1, \ldots, Q_d) \in (\mathbb{Z}^{m_{\text{int}} \times m_{\text{int}}})^d$. In this setup, each component of Ψ_Q is a quadratic function. For a given Q, $\mathbf{Z}$ is sampled according to the probability distribution $\pi_{\mathbf{Z}}$ and we compute $\mathbf{X} = \Psi_Q(\mathbf{Z})$ to generate the dataset. In practice we take $\pi_{\mathbf{Z}} = \mathcal{U}([-2,2]^{m_{\text{int}}})$ and the coefficients of Q are chosen randomly in $\{-5, \ldots, 5\}$. For $d = 8$ and $m_{\text{int}} = 3$

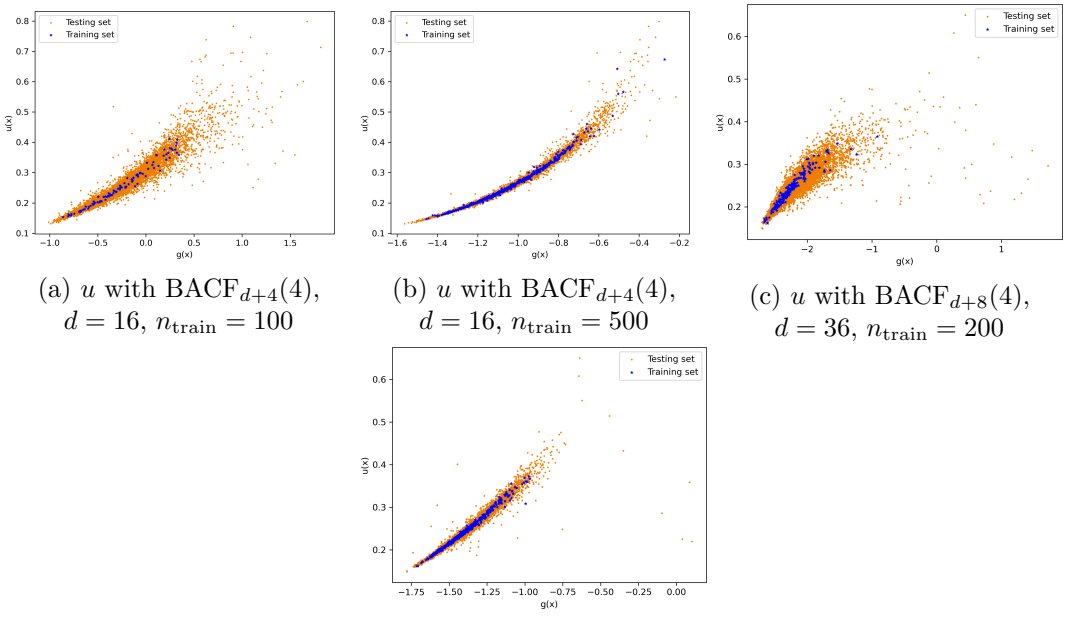
$m = 1$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 100$	BACF_IS(6)	1632	$(1.25 \pm 0.59) \times 10^{-3}$	5.18 ± 1.11	12.97 ± 2.79	8.43 ± 2.77
	BACF_IS$_{d+4}$(4)	1680	$(\mathbf{6.33} \pm 1.90) \times \mathbf{10^{-4}}$	$\mathbf{3.73} \pm 0.56$	$\mathbf{9.34} \pm 1.40$	$\mathbf{5.69} \pm 1.26$
	BACF_FS(6)	1632	$(1.27 \pm 0.42) \times 10^{-3}$	5.27 ± 0.86	13.20 ± 2.16	8.53 ± 2.08
	BACF_FS$_{d+4}$(4)	1680	$(2.56 \pm 0.05) \times 10^{-2}$	7.54 ± 0.07	18.89 ± 1.96	13.14 ± 1.40
	PFM	152	$(4.76 \pm 0.27) \times 10^{-3}$	10.34 ± 0.29	25.88 ± 0.74	20.10 ± 0.96
$n_{\text{train}} = 500$	BACF_IS(6)	1632	$(1.81 \pm 0.56) \times 10^{-4}$	2.00 ± 0.31	5.01 ± 0.77	2.19 ± 0.42
	BACF_IS$_{d+4}$(4)	1680	$(\mathbf{1.32} \pm 0.55) \times \mathbf{10^{-4}}$	$\mathbf{1.68} \pm 0.35$	$\mathbf{4.22} \pm 0.87$	$\mathbf{1.92} \pm 0.44$
	BACF_FS(6)	1632	$(4.47 \pm 5.51) \times 10^{-4}$	2.83 ± 1.41	7.10 ± 3.54	3.61 ± 3.84
	BACF_FS$_{d+4}$(4)	1680	$(3.65 \pm 2.98) \times 10^{-4}$	2.74 ± 0.83	6.86 ± 2.09	3.70 ± 0.42
	PFM	152	$(4.74 \pm 0.50) \times 10^{-3}$	10.34 ± 0.55	25.90 ± 1.39	19.79 ± 0.87
$n_{\text{train}} = 2500$	NLL	-	-	6.26	-	12.71
	DRiLLS	-	-	2.19	-	2.72

$m = 2$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 100$	BACF_IS(6)	1632	$(1.27 \pm 0.37) \times 10^{-3}$	5.29 ± 0.76	13.24 ± 1.91	8.28 ± 1.75
	BACF_IS$_{d+4}$(4)	1680	$(\mathbf{1.18} \pm 0.55) \times \mathbf{10^{-3}}$	$\mathbf{5.05} \pm 1.02$	$\mathbf{12.66} \pm 2.55$	$\mathbf{7.26} \pm 1.35$
	BACF_FS(6)	1632	$(1.28 \pm 0.32) \times 10^{-3}$	5.31 ± 0.65	13.31 ± 1.63	8.33 ± 1.28
	BACF_FS$_{d+4}$(4)	1680	$(4.25 \pm 1.47) \times 10^{-3}$	9.64 ± 1.57	24.15 ± 3.93	16.15 ± 1.80
	PFM	304	$(4.65 \pm 0.27) \times 10^{-3}$	10.21 ± 0.29	25.88 ± 0.73	19.78 ± 1.00
$n_{\text{train}} = 500$	BACF_IS(6)	1632	$(2.73 \pm 4.21) \times 10^{-4}$	2.18 ± 1.17	5.47 ± 2.93	3.46 ± 3.42
	BACF_IS$_{d+4}$(4)	1680	$(\mathbf{1.42} \pm 1.20) \times \mathbf{10^{-4}}$	$\mathbf{1.70} \pm 0.55$	$\mathbf{4.25} \pm 1.38$	$\mathbf{2.39} \pm 1.73$
	BACF_FS(6)	1632	$(2.31 \pm 1.42) \times 10^{-4}$	2.20 ± 0.58	5.52 ± 1.44	2.57 ± 0.40
	BACF_FS$_{d+4}$(4)	1680	$(4.16 \pm 1.88) \times 10^{-4}$	2.99 ± 0.61	7.50 ± 1.53	3.95 ± 0.46
	PFM	304	$(4.74 \pm 0.28) \times 10^{-3}$	10.31 ± 0.31	25.83 ± 0.78	19.83 ± 0.70
$n_{\text{train}} = 2500$	NLL	-	-	6.66	-	13.45
	DRiLLS	-	-	-	-	-

Table 6: Approximation errors for u on $\mathcal{X} = [0.1, 10]^{16}$ for the thermal block problem. The results for DRiLLS and NLL are copied from (Teng et al., 2021, Table 7).

$m = 1$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 200$	BACF_IS(6)	7992	$(\mathbf{3.00} \pm 0.64) \times \mathbf{10^{-4}}$	$\mathbf{3.44} \pm 0.34$	$\mathbf{6.91} \pm 0.69$	$\mathbf{4.73} \pm 0.38$
	$\text{BACF\_IS}_{d+8}(4)$	7920	$(3.13 \pm 0.69) \times 10^{-4}$	3.51 ± 0.35	7.06 ± 0.71	4.88 ± 0.40
	BACF_FS(6)	7992	$(3.61 \pm 0.32) \times 10^{-4}$	3.78 ± 0.16	7.60 ± 0.33	5.34 ± 0.22
	$\text{BACF\_FS}_{d+8}(4)$	7920	$(1.03 \pm 0.07) \times 10^{-3}$	6.40 ± 0.21	12.87 ± 0.42	9.72 ± 0.33
	PFM	702	$(2.04 \pm 1.12) \times 10^{-3}$	8.82 ± 1.87	17.72 ± 3.75	13.04 ± 0.34
$n_{\text{train}} = 500$	BACF_IS(6)	7992	$(8.48 \pm 4.67) \times 10^{-5}$	1.79 ± 0.40	3.60 ± 0.80	2.32 ± 0.98
	$\text{BACF\_IS}_{d+8}(4)$	7920	$(\mathbf{5.48} \pm 1.20) \times \mathbf{10^{-5}}$	$\mathbf{1.47} \pm 0.16$	$\mathbf{2.95} \pm 0.32$	$\mathbf{1.76} \pm 0.27$
	BACF_FS(6)	7992	$(1.09 \pm 0.59) \times 10^{-4}$	2.04 ± 0.45	4.09 ± 0.91	2.59 ± 1.08
	$\text{BACF\_FS}_{d+8}(4)$	7920	$(4.68 \pm 0.30) \times 10^{-4}$	4.31 ± 0.14	8.67 ± 0.27	6.34 ± 0.19
	PFM	702	$(1.74 \pm 0.02) \times 10^{-3}$	8.31 ± 0.52	16.71 ± 0.10	12.94 ± 0.13
$m = 2$						
Points	Structure	#Param	MSE%	NRMSE%	RL_2%	RL_1%
$n_{\text{train}} = 200$	BACF_IS(6)	7992	$(3.66 \pm 0.93) \times 10^{-4}$	3.79 ± 0.46	7.61 ± 0.93	5.14 ± 0.77
	$\text{BACF\_IS}_{d+8}(4)$	7920	$(\mathbf{3.54} \pm 0.62) \times \mathbf{10^{-4}}$	$\mathbf{3.74} \pm 0.31$	$\mathbf{7.52} \pm 0.62$	$\mathbf{5.06} \pm 0.36$
	BACF_FS(6)	7992	$(3.80 \pm 0.62) \times 10^{-4}$	3.87 ± 0.30	7.79 ± 0.61	5.41 ± 0.30
	$\text{BACF\_FS}_{d+8}(4)$	7920	$(1.61 \pm 0.07) \times 10^{-3}$	7.87 ± 1.48	15.82 ± 2.98	11.89 ± 2.31
	PFM	1404	$(1.68 \pm 0.10) \times 10^{-3}$	8.16 ± 0.26	16.40 ± 0.51	12.72 ± 0.49
$n_{\text{train}} = 500$	BACF_IS(6)	7992	$(1.32 \pm 1.44) \times 10^{-4}$	2.11 ± 0.90	4.23 ± 1.81	2.73 ± 1.92
	$\text{BACF\_IS}_{d+8}(4)$	7920	$(1.02 \pm 0.85) \times 10^{-4}$	1.91 ± 0.64	3.84 ± 1.28	2.58 ± 1.41
	BACF_FS(6)	7992	$(\mathbf{7.98} \pm 0.82) \times \mathbf{10^{-5}}$	$\mathbf{1.78} \pm 0.09$	$\mathbf{3.58} \pm 0.18$	$\mathbf{2.27} \pm 0.11$
	$\text{BACF\_FS}_{d+8}(4)$	7920	$(5.02 \pm 0.39) \times 10^{-4}$	4.46 ± 0.17	8.98 ± 0.35	6.55 ± 0.31
	PFM	1404	$(1.70 \pm 0.06) \times 10^{-3}$	8.23 ± 0.15	16.53 ± 0.29	12.80 ± 0.24

Table 7: Approximation errors of u on $\mathcal{X} = [0.1, 10]^{36}$ for the thermal block problem



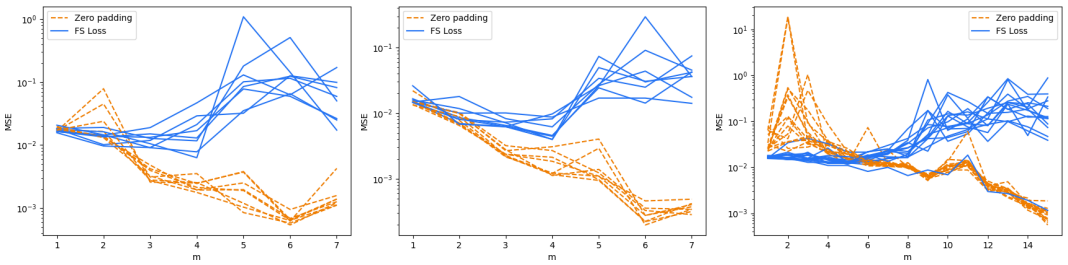
(a) u with $\text{BACF}_{d+4}(4)$, $d = 16$, $n_{\text{train}} = 100$

(b) u with $\text{BACF}_{d+4}(4)$, $d = 16$, $n_{\text{train}} = 500$

(c) u with $\text{BACF}_{d+8}(4)$, $d = 36$, $n_{\text{train}} = 200$

(d) u with $\text{BACF}_{d+8}(4)$, $d = 36$, $n_{\text{train}} = 500$

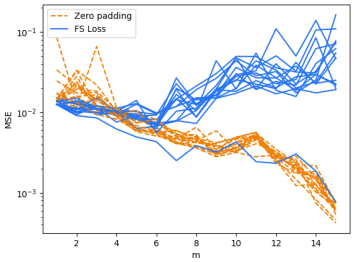
Figure 2: Scatter plot $\{(g(x^i), u(x^i))\}_{i \geq 1}$ for the thermal block problem using a random testing set of $n_{\text{test}} = 10^4$ points on $\mathcal{X} = [0.1, 10]^d$ when $m = 1$



(a) Q_8 datasets with $n_{train} = 100$

(b) Q_8 datasets with $n_{train} = 500$

(c) Q_{16} datasets with $n_{train} = 100$



(d) Q_{16} datasets with $n_{train} = 500$

Figure 3: Reconstruction errors according to m using either the reconstruction loss with zero padding $\mathcal{L}$ (Zero padding) or $\mathcal{J}^{\mathrm{FS}}_{m,\lambda}$(FS Loss). We use BACF(4) with $\lambda = 10^{-2}$.

(respectively $d = 16$ and $m_{\mathrm{int}} = 5$), we set $Q = Q_8$ (respectively $Q = Q_{16}$) and we generate the datasets as described above. The values of Q_8 and Q_{16} are given in Appendix F. Following Section 5.2, we build $g = (\varphi_1, \ldots, \varphi_m)$ either by minimizing $\mathcal{J}^{\mathrm{FS}}_{m,\lambda}(\varphi)$ (`FS Loss`) or the reconstruction loss $\mathcal{L}_m(\varphi)$ as defined in (29) (`Zero padding`). We train g and f on 15 training sets and we compute the error $\mathbb{E}(\|\mathbf{X} - f \circ g(\mathbf{X})\|^2)$ on a test set of size $n_{\mathrm{test}} = 10^4$.

Figure 3 plots the test error as a function of m for the 15 training sets, each line corresponding to a different training set. For $m \leq m_{\mathrm{int}}$, `FS Loss` yields similar results compared to `Zero padding` but with significantly less variance over the 15 trials. For $m > m_{\mathrm{int}}$, however, the performances of `FS Loss` degenerate and `Zero padding` yields much better results. As already pointed out in (Nguyen et al., 2019), the reconstruction error of `Zero padding` continues to decrease with m even when $m > m_{\mathrm{int}}$ exceeds the intrinsic dimension. Further research needs to be conducted to understand why this is not the case with `FS Loss`.

8 Conclusion

In this paper, we propose and compare different strategies for learning nonlinear features in a supervised learning framework where gradients of the high-dimensional function to be approximated are available. By seeking the feature map as the first components of a diffeomorphism, we establish bounds for the reconstruction error using Poincaré Inequalities applied either in the input space or in the feature space. Our computational experiments demonstrate that the approach based on the augmented input space not only exceeds

performances of the feature space method but also outperforms all comparable state-of-the-art techniques found in existing literature. By extending our method to vector-valued functions, we are able to learn autoencoders for unsupervised learning tasks. However, our numerical experiments indicate that the resulting method performs suboptimally compared to existing approaches, necessitating further investigation. In practical implementation, we learn the diffeomorphism (and consequently, the feature map) by employing block affine coupling flows, a class of invertible neural networks easily implementable with modern machine learning libraries. Future research is required to determine the optimal choice for the latent dimension m and the augmented dimension k a priori. The control of Poincaré constant, moreover, remains an open and challenging theoretical question.

References

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

Dominique Bakry and Michel Émery. Diffusions hypercontractives. In *Séminaire de Probabilités XIX 1983/84: Proceedings*, pages 177–206. Springer, 2006.

Dominique Bakry, Ivan Gentil, Michel Ledoux, et al. *Analysis and geometry of Markov diffusion operators*, volume 103. Springer, 2014.

Ricardo Baptista, Youssef Marzouk, and Olivier Zahm. Gradient-based data and parameter dimension reduction for bayesian models: an information theoretic perspective. *arXiv preprint arXiv:2207.08670*, 2022.

Ricardo Baptista, Youssef Marzouk, and Olivier Zahm. On the representation and learning of monotone triangular transport maps. *Foundations of Computational Mathematics*, pages 1–46, 2023.

Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018.

Daniele Bigoni, Youssef Marzouk, Clémentine Prieur, and Olivier Zahm. Nonlinear dimension reduction for surrogate modeling using gradient information. *Information and Inference: A Journal of the IMA*, 11(4):1597–1639, 2022.

Paul G Constantine, Eric Dow, and Qiqi Wang. Active subspace methods in theory and practice: applications to kriging surfaces. *SIAM Journal on Scientific Computing*, 36(4): A1500–A1524, 2014.

Tiangang Cui, James Martin, Youssef M Marzouk, Antti Solonen, and Alessio Spantini. Likelihood-informed dimension reduction for nonlinear inverse problems. *Inverse Problems*, 30(11):114015, 2014.

Sébastien Da Veiga, Fabrice Gamboa, Bertrand Iooss, and Clémentine Prieur. *Basics and Trends in Sensitivity Analysis*. Society for Industrial and Applied Mathematics,

Philadelphia, PA, 2021. doi: 10.1137/1.9781611976694. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611976694.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *ICLR workshop*, 2015.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *ICLR*, 2017.

David L Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10): 5591–5596, 2003.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *Advances in neural information processing systems*, 32, 2019.

A. Ern and J.L. Guermond. *Theory and Practice of Finite Elements*. Applied Mathematical Sciences. Springer New York, 2004.

Andreas Griewank et al. On automatic differentiation. *Mathematical Programming: recent developments and applications*, 6(6):83–107, 1989.

Anthony Gruber, Max Gunzburger, Lili Ju, Yuankai Teng, and Zhu Wang. Nonlinear level set learning for function approximation on sparse data with applications to parametric differential equations. *arXiv preprint arXiv:2104.14072*, 2021.

Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. scikit-optimize/scikit-optimize, 2022. URL https://doi.org/10.5281/zenodo.6451894.

Marian Hristache, Anatoli Juditsky, Jorg Polzehl, and Vladimir Spokoiny. Structure adaptive approach for dimension reduction. *Annals of Statistics*, pages 1537–1566, 2001a.

Marian Hristache, Anatoli Juditsky, and Vladimir Spokoiny. Direct estimation of the index coefficient in a single-index model. *Annals of Statistics*, pages 595–623, 2001b.

Chin-Wei Huang, Laurent Dinh, and Aaron Courville. Augmented normalizing flows: Bridging the gap between generative flows and latent variable models. *arXiv preprint arXiv:2002.07101*, 2020.

Isao Ishikawa, Takeshi Teshima, Koichi Tojo, Kenta Oono, Masahiro Ikeda, and Masashi Sugiyama. Universal approximation property of invertible neural networks. *arXiv preprint arXiv:2204.07415*, 2022.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, 2021.

Matthew TC Li, Youssef Marzouk, and Olivier Zahm. Principal feature detection via ϕ-sobolev inequalities. *arXiv preprint arXiv:2305.06172*, 2023.

Junlong Lyu, Zhitang Chen, Chang Feng, Wenjing Cun, Shengyu Zhu, Yanhui Geng, ZHIJIE XU, and Chen Yongwei. Para-cflows: C^k-universal diffeomorphism approximators as superior neural surrogates. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 28829–28841. Curran Associates, Inc., 2022.

The-Gia Leo Nguyen, Lynton Ardizzone, and Ullrich Köthe. Training invertible neural networks as autoencoders. pages 442–455, 2019.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *The Journal of Machine Learning Research*, 22(1):2617–2680, 2021.

Mario Teixeira Parente, Jonas Wallin, and Barbara Wohlmuth. Generalized bounds for active subspaces. *Electronic Journal of Statistics*, 14(1):917 – 943, 2020. doi: 10.1214/20-EJS1684. URL `https://doi.org/10.1214/20-EJS1684`.

R-E Plessix. A review of the adjoint-state method for computing the gradient of a functional with geophysical applications. *Geophysical Journal International*, 167(2):495–503, 2006.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

Francesco Romor, Marco Tezzele, Andrea Lario, and Gianluigi Rozza. Kernel-based active subspaces with application to computational fluid dynamics parametric problems using the discontinuous galerkin method. *International Journal for Numerical Methods in Engineering*, 123(23):6000–6027, 2022.

Alexander M Samarov. Exploring regression structure using nonparametric functional estimation. *Journal of the American Statistical Association*, 88(423):836–847, 1993.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Artificial Neural NetworksICANN'97: 7th International Conference Lausanne, Switzerland, October 8–10, 1997 Proceeedings*, pages 583–588. Springer, 2005.

Joshua B Tenenbaum, Vin de Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Yuankai Teng, Zhu Wang, Lili Ju, Anthony Gruber, and Guannan Zhang. Level set learning with pseudo-reversible neural networks for nonlinear dimension reduction in function approximation. *arXiv preprint arXiv:2112.01438*, 2021.

Takeshi Teshima, Isao Ishikawa, Koichi Tojo, Kenta Oono, Masahiro Ikeda, and Masashi Sugiyama. Coupling-based invertible neural networks are universal diffeomorphism approximators. *Advances in Neural Information Processing Systems*, 33:3362–3373, 2020.

Olivier Zahm, Paul G Constantine, Clémentine Prieur, and Youssef M Marzouk. Gradient-based dimension reduction of multivariate vector-valued functions. *SIAM Journal on Scientific Computing*, 42(1):A534–A558, 2020.

Olivier Zahm, Tiangang Cui, Kody Law, Alessio Spantini, and Youssef Marzouk. Certified dimension reduction in nonlinear bayesian inverse problems. *Mathematics of Computation*, 91(336):1789–1835, 2022.

Guannan Zhang, Jiaxin Zhang, and Jacob Hinkle. Learning nonlinear level sets for dimensionality reduction in function approximation. *Advances in Neural Information Processing Systems*, 32, 2019a.

Han Zhang, Xi Gao, Jacob Unterman, and Tom Arodz. Approximation capabilities of neural ordinary differential equations. *arXiv preprint arXiv:1907.12998*, 2(4):3–1, 2019b.

Appendix A. Proof of Proposition 2

The fact that $u = f \circ g$ for some $f \in \mathcal{C}^1(\mathbb{R}^m; \mathbb{R})$ implies $\mathcal{J}^{\mathrm{IS}}(g) = 0$ is a direct consequence of the chain-rule. To show the reciprocal, let $\varphi \in \mathcal{D}$ be a diffeomorphism such that $g(x) = (\varphi_1(x), \dots, \varphi_m(x))$ and assume that $\mathcal{J}^{\mathrm{IS}}(g) = 0$. The gradient of $u = (u \circ \varphi^{-1}) \circ \varphi$ is $\nabla u(x) = \nabla\varphi(x)^\top \nabla(u \circ \varphi^{-1}) \circ \varphi(x)$ and it decomposes as

$$\begin{aligned}\nabla u(x) &= [\nabla\varphi_1(x), \dots, \nabla\varphi_d(x)] \begin{pmatrix} \partial_1(u \circ \varphi^{-1}) \circ \varphi(x) \\ \vdots \\ \partial_d(u \circ \varphi^{-1}) \circ \varphi(x) \end{pmatrix} \\ &= \nabla g(x)^\top \nabla_m (u \circ \varphi^{-1}) \circ \varphi(x) + \nabla\varphi_\perp(x)^\top \nabla_\perp (u \circ \varphi^{-1}) \circ \varphi(x), \end{aligned} \tag{32}$$

where $\varphi_\perp(x) = (\varphi_{m+1}(x), \dots, \varphi_d(x))$, $\nabla_m = (\partial_1, \dots, \partial_m)$ and $\nabla_\perp = (\partial_{m+1}, \dots, \partial_d)$. Since $\varphi \in \mathcal{D}$ is a diffeomorphism, its Jacobian has full rank everywhere so that $\mathrm{range}(\nabla g(x)^\top) \oplus \mathrm{range}(\nabla\varphi_\perp(x)^\top) = \mathbb{R}^d$. Therefore, because $\mathcal{J}^{\mathrm{IS}}(g) = 0$ implies $\nabla u(\mathbf{X}) \in \mathrm{range}(\nabla g(\mathbf{X})^\top)$ almost surely, relation (32) yields $\nabla_\perp(u \circ \varphi^{-1}) \circ \varphi(\mathbf{X}) = 0$ almost surely. By continuity of $u \circ \varphi^{-1} : \mathbb{R}^d \to \mathbb{R}$ and because $\mathrm{supp}(\varphi(\mathbf{X})) = \mathbb{R}^d$, we deduce that $\nabla_\perp(u \circ \varphi^{-1}) = 0$ everywhere so that $z_\perp \mapsto u \circ \varphi^{-1}(z_m, z_\perp)$ is constant for any z_m. Thus, for $f \in \mathcal{C}^1(\mathbb{R}^m; \mathbb{R})$ being defined as $f(z_m) = u \circ \varphi^{-1}(z_m, 0)$, we have $f \circ g(x) = u \circ \varphi^{-1}(g(x), \varphi_\perp(x)) = u \circ \varphi^{-1}(\varphi_1(x), \dots, \varphi_d(x)) = u(x)$.

It remains to show (11). Recall that the function f which realizes the minimium in (3) is the conditional expectation so that $\mathcal{E}(g) = \mathbb{E}[(u(\mathbf{X}) - \mathbb{E}[u(\mathbf{X})|g(\mathbf{X})])^2]$. Using the tower property of the conditional expectation, we can write

$$\begin{aligned}\mathcal{E}(g) &= \mathbb{E}\left[\mathbb{E}[(u(\mathbf{X}) - \mathbb{E}[u(\mathbf{X})|g(\mathbf{X})])^2|g(\mathbf{X})]\right] \\ &\le \mathbb{E}\left[\mathbb{C}(\mathbf{X}|g(\mathbf{X}))\mathbb{E}[\|\nabla u_{|\mathcal{M}(\mathbf{X})}(\mathbf{X})\|^2|g(\mathbf{X})]\right] \\ &\le \mathbb{C}^{\mathrm{IS}}(\mathbf{X}|\mathcal{G}_m)\mathbb{E}\left[\mathbb{E}[\|\nabla u_{|\mathcal{M}(\mathbf{X})}(\mathbf{X})\|^2|g(\mathbf{X})]\right] \\ &= \mathbb{C}^{\mathrm{IS}}(\mathbf{X}|\mathcal{G}_m)\mathbb{E}[\|\nabla u_{|\mathcal{M}(\mathbf{X})}(\mathbf{X})\|^2],\end{aligned}$$

where we recall that $\mathcal{M}(\mathbf{X}) = \{x \in \mathcal{X} \text{ s.t. } g(x) = g(\mathbf{X})\}$. The above first inequality is the Poincaré Inequality (8) applied on $\mathbf{Y} = \mathbf{X}|g(\mathbf{X})$ as in the left column in Table 1, and the second inequality is a consequence of the definition of $\mathbb{C}^{\mathrm{IS}}(\mathbf{X}|\mathcal{G}_m)$. Finally, by (10), we deduce that $\mathcal{E}(g) \le \mathbb{C}^{\mathrm{IS}}(\mathbf{X}|\mathcal{G}_m)\mathcal{J}_m^{\mathrm{IS}}(g)$ which concludes the proof.

Appendix B. Proof of Proposition 4

The fact that $u = f \circ g$ for some $f \in \mathcal{C}^1(\mathbb{R}^m; \mathbb{R})$ implies $\mathcal{J}_m^{\mathrm{FS}}(\varphi) = 0$ is a consequence of the chain-rule

$$\nabla u(x) = \nabla g(x)^\top \nabla f(g(x)) = \nabla\varphi(x)^\top \begin{pmatrix} \nabla f(g(x)) \\ 0 \end{pmatrix},$$

where we recall that $\nabla\varphi(x)^\top = [\nabla g(x)^\top, \nabla\varphi_\perp(x)^\top]$ is everywhere invertible. Therefore, $\nabla\varphi(x)^{-\top}\nabla u(x) = \left(\begin{smallmatrix} \nabla f(g(x)) \\ 0 \end{smallmatrix}\right)$ so that $\mathcal{J}_m^{\mathrm{FS}}(\varphi) = 0$. To show the reciprocal, note that $\mathcal{J}_m^{\mathrm{FS}}(\varphi) = 0$ implies, by (14), that $\nabla_\perp(u \circ \varphi^{-1}) \circ \varphi(\mathbf{X}) = 0$ almost surely. By continuity of $u \circ \varphi^{-1}$ and because $\mathrm{supp}(\varphi(\mathbf{X})) = \mathbb{R}^d$, we deduce that $z_\perp \mapsto u \circ \varphi^{-1}(z_m, z_\perp)$ is constant for any $z_m \in \mathbb{R}^m$ and therefore that $u = f \circ g$ for $f(z_m) = u \circ \varphi^{-1}(z_m, 0)$.

It remains to show (16). By the change of variable $\mathbf{Z} = \varphi(\mathbf{X})$, we can write $g(\mathbf{X}) = \mathbf{Z}_m$ so that the reconstruction error writes $\mathcal{E}(g) = \mathbb{E}[(u \circ \varphi^{-1}(\mathbf{Z}) - \mathbb{E}[u \circ \varphi^{-1}(\mathbf{Z})|\mathbf{Z}_m])^2]$. Using the tower property of the conditional expectation, we can write

$$\begin{aligned}
\mathcal{E}(g) &= \mathbb{E}\left[\mathbb{E}[(u \circ \varphi^{-1}(\mathbf{Z}) - \mathbb{E}[u \circ \varphi^{-1}(\mathbf{Z})|\mathbf{Z}_m])^2|\mathbf{Z}_m]\right] \\
&\leq \mathbb{E}\left[\mathbb{C}(\mathbf{Z}|\mathbf{Z}_m)\mathbb{E}[\|\nabla_\perp (u \circ \varphi^{-1})(\mathbf{Z})\|^2|\mathbf{Z}_m)]\right] \\
&\leq \mathbb{C}_m^{\mathrm{FS}}(\mathbf{X}|\mathcal{D})\mathbb{E}\left[\mathbb{E}[\|\nabla_\perp (u \circ \varphi^{-1})(\mathbf{Z})\|^2|\mathbf{Z}_m)]\right] \\
&= \mathbb{C}_m^{\mathrm{FS}}(\mathbf{X}|\mathcal{D})\mathcal{J}_m^{\mathrm{FS}}(\varphi),
\end{aligned}$$

where, for the last equality, we used Equation (14). The above first inequality is the Poincaré Inequality (8) applied on $\mathbf{Y} = \mathbf{Z}|\mathbf{Z}_m$ as in the right column in Table 1, and the second inequality is a consequence of the definition of $\mathbb{C}_m^{\mathrm{FS}}(\mathbf{X}|\mathcal{D})$. This concludes the proof.

Appendix C. Proof of Proposition 6

Let $P_m = \mathrm{diag}(1, \dots, 1, 0, \dots, 0)$ be such that $P_m v = (v_1, \dots, v_m, 0, \dots, 0)$ for $v \in \mathbb{R}^d$. Because $M(\mathbf{X}) = \nabla\varphi(\mathbf{X})^{-1}\nabla\varphi(\mathbf{X})^{-\top}$, we can write

$$\begin{aligned}
\mathcal{J}_m^{\mathrm{FS}}(\varphi) &= \mathbb{E}[\|(\nabla\varphi(\mathbf{X})^{-\top}\nabla u(\mathbf{X}))_\perp\|^2] \\
&= \mathbb{E}[\|(I_d - P_m)\nabla\varphi(\mathbf{X})^{-\top}\nabla u(\mathbf{X})\|^2] \\
&= \mathbb{E}[\|\nabla u(\mathbf{X}) - \nabla\varphi(\mathbf{X})^\top P_m \nabla\varphi(\mathbf{X})^{-\top}\nabla u(\mathbf{X})\|^2_{M(\mathbf{X})}] \\
&= \mathbb{E}[\|\nabla u(\mathbf{X}) - \Pi^{M(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2_{M(\mathbf{X})}]
\end{aligned}$$

where we used the fact that $\Pi^{M(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)} = \nabla\varphi(\mathbf{X})^\top P_m \nabla\varphi(\mathbf{X})^{-\top}$ is the $\langle\cdot,\cdot\rangle_{M(\mathbf{X})}$-orthogonal projector onto $\mathrm{range}(\nabla g(\mathbf{X})^\top) = \mathrm{range}(\nabla\varphi(\mathbf{X})^\top P_m)$. This shows (19).

To show (20), let us notice that by definition of orthogonal projectors, we can write

$$\begin{aligned}
\mathcal{J}_m^{\mathrm{FS}}(\varphi) &= \mathbb{E}[\|\nabla u(\mathbf{X}) - \Pi^{M(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2_{M(\mathbf{X})}] \\
&\leq \mathbb{E}[\|\nabla u(\mathbf{X}) - \Pi_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2_{M(\mathbf{X})}] \\
&\leq \mathbb{E}\left[\left(\sup_{v\in\mathbb{R}^d}\frac{\|v\|^2_{M(\mathbf{X})}}{\|v\|^2}\right)\|\nabla u(\mathbf{X}) - \Pi_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2\right] \\
&\leq \sup_{x\in\mathcal{X},v\in\mathbb{R}^d}\frac{\|v\|^2_{M(x)}}{\|v\|^2}\mathcal{J}_m^{\mathrm{IS}}(g).
\end{aligned}$$

Furthermore we can write

$$\sup_{x\in\mathcal{X},v\in\mathbb{R}^d}\frac{\|v\|_{M(x)}}{\|v\|} = \sup_{z,v\in\mathbb{R}^d}\frac{\|(\nabla\varphi(\varphi^{-1}(z)))^{-1}v\|}{\|v\|} = \sup_{z,v\in\mathbb{R}^d}\frac{\|\nabla\varphi^{-1}(z)v\|}{\|v\|} \leq \mathrm{Lip}(\varphi^{-1})$$

where we used the fact that the norm of the gradient of a function is bounded by the Lipschitz constant of that function. This yields the right-hand side of (20). Similarily we

have

$$\begin{aligned}\mathcal{J}_m^{\mathrm{IS}}(g) &= \mathbb{E}[\|\nabla u(\mathbf{X}) - \Pi_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2] \\ &\leq \mathbb{E}[\|\nabla u(\mathbf{X}) - \Pi^{M(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2] \\ &\leq \mathbb{E}\left[\left(\sup_{v\in\mathbb{R}^d}\frac{\|v\|^2}{\|v\|^2_{M(\mathbf{X})}}\right)\|\nabla u(\mathbf{X}) - \Pi^{M(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2_{M(\mathbf{X})}\right] \\ &\leq \sup_{x\in\mathcal{X},v\in\mathbb{R}^d}\frac{\|v\|^2}{\|v\|^2_{M(x)}}\mathcal{J}_m^{\mathrm{FS}}(\varphi).\end{aligned}$$

and

$$\sup_{x\in\mathcal{X},v\in\mathbb{R}^d}\frac{\|v\|}{\|v\|_{M(x)}} = \sup_{x\in\mathcal{X},v\in\mathbb{R}^d}\frac{\|v\|}{\|\nabla\varphi(x)^{-1}v\|} = \sup_{x\in\mathcal{X},v\in\mathbb{R}^d}\frac{\|\nabla\varphi(x)v\|}{\|v\|} \leq \mathrm{Lip}(\varphi).$$

This yields the left-hand side of (20) and concludes the proof.

Appendix D. Proof of Proposition 7

The proof of Proposition 7 is similar to the one of Proposition 6. Let $P_m = \mathrm{diag}(1,\dots,1,0,\dots,0)$ be such that $P_m v = (v_1,\dots,v_m,0,\dots,0)$ for $v\in\mathbb{R}^d$. By definition (21) of $\mathcal{J}_m^{\mathrm{NLL}}(\varphi)$, we can write

$$\begin{aligned}\mathcal{J}_m^{\mathrm{NLL}}(\varphi) &= \mathbb{E}\left[\|(I_d - P_m)\mathrm{diag}(\|J_i(\mathbf{X})\|^{-1})\nabla\varphi(\mathbf{X})^{-\top}\nabla u(\mathbf{X})\|^2\right] \\ &= \mathbb{E}\left[\|\nabla u(\mathbf{X}) - \nabla\varphi(\mathbf{X})^\top\mathrm{diag}(\|J_i(\mathbf{X})\|)P_m\mathrm{diag}(\|J_i(\mathbf{X})\|^{-1})\nabla\varphi(\mathbf{X})^{-\top}\nabla u(\mathbf{X})\|^2_{D(\mathbf{X})}\right] \\ &= \mathbb{E}\left[\|\nabla u(\mathbf{X}) - \nabla\varphi(\mathbf{X})^\top P_m\nabla\varphi(\mathbf{X})^{-\top}\nabla u(\mathbf{X})\|^2_{D(\mathbf{X})}\right] \\ &= \mathbb{E}\left[\|\nabla u(\mathbf{X}) - \Pi^{D(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)}\nabla u(\mathbf{X})\|^2_{D(\mathbf{X})}\right],\end{aligned}$$

where we use the fact that $\Pi^{D(\mathbf{X})}_{\mathrm{range}(\nabla g(\mathbf{X})^\top)} = \nabla\varphi(\mathbf{X})^\top P_m\nabla\varphi(\mathbf{X})^{-\top}$ is the $\langle\cdot,\cdot\rangle_{D(\mathbf{X})}$-orthogonal projector onto $\mathrm{range}(\nabla\varphi(\mathbf{X})^\top P_m) = \mathrm{range}(\nabla g(\mathbf{X})^\top)$.

Appendix E. Proof of Proposition 9

Notice that the Jacobian of the function defined by (24) reads

$$\nabla\varphi(x,\xi) = \begin{pmatrix}\nabla g(x) & I_m \\ I_d & 0_{d\times m}\end{pmatrix}, \quad \text{and} \quad \nabla\varphi^{-1}(z_1,z_2) = \begin{pmatrix}0_{d\times m} & I_d \\ I_m & -\nabla g(z_2)\end{pmatrix}.$$

Using the relation $\nabla\varphi(x,\xi)^{-1} = \nabla\varphi^{-1}(\varphi(x,\xi))$, we obtain

$$\mathcal{J}_{m,m}^{\mathrm{FS}}(\varphi) = \mathbb{E}\left[\left\|\left(\begin{pmatrix}0_{m\times d} & I_m \\ I_d & -\nabla g(x)^\top\end{pmatrix}\begin{pmatrix}\nabla u(\mathbf{X}) \\ 0_m\end{pmatrix}\right)_\perp\right\|^2\right] = \mathbb{E}\left[\|\nabla u(\mathbf{X})\|^2\right].$$

Furthermore, because $\nabla g(x,\xi) = (\nabla g(x), I_m)$, we can write

$$\Pi_{\mathrm{range}(\nabla g(x,\xi)^\top)} = \begin{pmatrix} \nabla g(x)^\top \\ I_m \end{pmatrix} (I_m + \nabla g(x)\nabla g(x)^\top)^{-1} \begin{pmatrix} \nabla g(x)^\top \\ I_m \end{pmatrix}^\top$$

so that, using Pythagorean Theorem, we obtain

$$\begin{aligned} \mathcal{J}^{\mathrm{IS}}_{m,m}(g) &= \mathbb{E}[\|\nabla u(\mathbf{X})\|^2] - \mathbb{E}\left[\begin{pmatrix} \nabla u(\mathbf{X}) \\ 0_m \end{pmatrix}^\top \Pi_{\mathrm{range}(\nabla g(\mathbf{X},\boldsymbol{\Xi})^\top)} \begin{pmatrix} \nabla u(\mathbf{X}) \\ 0_m \end{pmatrix} \right] \\ &= \mathbb{E}[\|\nabla u(\mathbf{X})\|^2] - \mathbb{E}\left[u(\mathbf{X})^\top \nabla g(\mathbf{X})^\top (I_m + \nabla g(\mathbf{X})\nabla g(\mathbf{X})^\top)^{-1} \nabla g(\mathbf{X}) \nabla u(\mathbf{X}) \right]. \end{aligned}$$

This concludes the proof.

Appendix F. Q_8 and Q_{16} values

$Q_8 = (Q_{8,1}, Q_{8,2}, \ldots, Q_{8,8}) \in (\mathbb{Z}^{3\times 3})^8$

$$Q_{8,1} = \begin{pmatrix} 4 & 0 & -5 \\ 0 & 2 & 1 \\ -3 & -5 & -3 \end{pmatrix} \quad Q_{8,2} = \begin{pmatrix} -1 & 3 & 2 \\ -1 & 0 & 1 \\ -3 & -2 & -3 \end{pmatrix}$$

$$Q_{8,3} = \begin{pmatrix} 4 & 4 & -3 \\ 1 & -4 & 0 \\ 1 & 3 & 0 \end{pmatrix} \quad Q_{8,4} = \begin{pmatrix} -4 & 3 & 4 \\ 1 & -5 & 0 \\ 3 & 1 & 0 \end{pmatrix}$$

$$Q_{8,5} = \begin{pmatrix} -2 & -2 & -3 \\ -1 & 1 & -5 \\ 1 & 2 & -1 \end{pmatrix} \quad Q_{8,6} = \begin{pmatrix} 4 & -2 & -5 \\ -5 & -5 & 2 \\ -5 & 1 & -2 \end{pmatrix}$$

$$Q_{8,7} = \begin{pmatrix} -4 & -3 & 0 \\ -5 & -1 & 1 \\ 4 & 4 & 2 \end{pmatrix} \quad Q_{8,8} = \begin{pmatrix} -2 & -3 & -1 \\ -5 & -5 & -1 \\ -1 & -3 & -2 \end{pmatrix}$$

$Q_{16} = (Q_{16,1}, Q_{16,2}, \ldots, Q_{16,16}) \in (\mathbb{Z}^{5\times 5})^{16}$

$$Q_{16,1} = \begin{pmatrix} -2 & -5 & -4 & 4 & -4 \\ 2 & 3 & -1 & -1 & 2 \\ 3 & -5 & -3 & 0 & -5 \\ 0 & -1 & -5 & -1 & -3 \\ 4 & 1 & 3 & -4 & 3 \end{pmatrix} \quad Q_{16,2} = \begin{pmatrix} 1 & 1 & -5 & -5 & -5 \\ -5 & 1 & 4 & -3 & -5 \\ 2 & -5 & 1 & -5 & 1 \\ -2 & -2 & -2 & 2 & -3 \\ 2 & -5 & -5 & -2 & -2 \end{pmatrix}$$

$$Q_{16,3} = \begin{pmatrix} -3 & -4 & -1 & -3 & -3 \\ 3 & 3 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 & 2 \\ -1 & -1 & 3 & 2 & -4 \\ 4 & 0 & 0 & -4 & 4 \end{pmatrix} \quad Q_{16,4} = \begin{pmatrix} -2 & 1 & 3 & 1 & 1 \\ -4 & -5 & 4 & -5 & -4 \\ 1 & 0 & -5 & 0 & 3 \\ 0 & 3 & -3 & 3 & 3 \\ 0 & -1 & -1 & -3 & 2 \end{pmatrix}$$

$$Q_{16,5} = \begin{pmatrix} -4 & 2 & -1 & 0 & 3 \\ -4 & 0 & -1 & -5 & 0 \\ -2 & 2 & -2 & 0 & 2 \\ 3 & -4 & 0 & 3 & 1 \\ 0 & -2 & 3 & 2 & -2 \end{pmatrix} \quad Q_{16,6} = \begin{pmatrix} 3 & 3 & 4 & 3 & 0 \\ -4 & -3 & -4 & 4 & -5 \\ -3 & 0 & -2 & 4 & 2 \\ -3 & -2 & -4 & -1 & -3 \\ -4 & -3 & 3 & 1 & -5 \end{pmatrix}$$

$$Q_{16,7} = \begin{pmatrix} 4 & 2 & 0 & -2 & 1 \\ 3 & 1 & 2 & 0 & -2 \\ 0 & 2 & 0 & -2 & 2 \\ -5 & 3 & 3 & -4 & 4 \\ -2 & -3 & -1 & 2 & 0 \end{pmatrix} \quad Q_{16,8} = \begin{pmatrix} 2 & 0 & -4 & 1 & 2 \\ 3 & -1 & 0 & 1 & -2 \\ -1 & 2 & 0 & 3 & 0 \\ -4 & -4 & 1 & 2 & 0 \\ 4 & -4 & 2 & 3 & 4 \end{pmatrix}$$

$$Q_{16,9} = \begin{pmatrix} 1 & -4 & -1 & 0 & -3 \\ 4 & -3 & -2 & 2 & 3 \\ 0 & 1 & 3 & 3 & -3 \\ 0 & -5 & -3 & -3 & 4 \\ 1 & 4 & 3 & -1 & -4 \end{pmatrix} \quad Q_{16,10} = \begin{pmatrix} -2 & -5 & 2 & -3 & 1 \\ -4 & 2 & -1 & 0 & -2 \\ -5 & -3 & -2 & -3 & -4 \\ -3 & -3 & -4 & 1 & -1 \\ 2 & -2 & 3 & -4 & -4 \end{pmatrix}$$

$$Q_{16,11} = \begin{pmatrix} 1 & 0 & -5 & 1 & 2 \\ 4 & 0 & 0 & 0 & -3 \\ 4 & -3 & -4 & 3 & 2 \\ 1 & 3 & 3 & -2 & 2 \\ -5 & -1 & 1 & 0 & 2 \end{pmatrix} \quad Q_{16,12} = \begin{pmatrix} 1 & -5 & 2 & -4 & 2 \\ -2 & -3 & -4 & 4 & 2 \\ 2 & -4 & -1 & 0 & -2 \\ 3 & 1 & -4 & 2 & 4 \\ 1 & -3 & 1 & 4 & 0 \end{pmatrix}$$

$$Q_{16,13} = \begin{pmatrix} 0 & 1 & -2 & 4 & -5 \\ -5 & 4 & 1 & 4 & -5 \\ 2 & 3 & 1 & -4 & 0 \\ 3 & 0 & 1 & -4 & 3 \\ 2 & -3 & -5 & 1 & -3 \end{pmatrix} \quad Q_{16,14} = \begin{pmatrix} 2 & -4 & 4 & 2 & 4 \\ -5 & 2 & -3 & 0 & 2 \\ 1 & -4 & -3 & 4 & 0 \\ 0 & -2 & -5 & -5 & 1 \\ -4 & -4 & -4 & -1 & -3 \end{pmatrix}$$

$$Q_{16,15} = \begin{pmatrix} -2 & -5 & 4 & -1 & 2 \\ 0 & 3 & -4 & -5 & 4 \\ 1 & 2 & -2 & 1 & 1 \\ -4 & 4 & -5 & 2 & 4 \\ -1 & 4 & -3 & 1 & 1 \end{pmatrix} \quad Q_{16,16} = \begin{pmatrix} -2 & -5 & -1 & 4 & 0 \\ 2 & -4 & -5 & 4 & 4 \\ -2 & 1 & 0 & 4 & 1 \\ 2 & -2 & -3 & 0 & 3 \\ 3 & 3 & 2 & -3 & 2 \end{pmatrix}$$