The Galerkin method beats Graph-Based Approaches for Spectral Algorithms

Vivien Cabannes
Meta AI

Francis Bach
INRIA, Ecole Normale Supérieure

Abstract

Historically, the machine learning community has derived spectral decompositions from graph-based approaches. We break with this approach and prove the statistical and computational superiority of the Galerkin method, which consists in restricting the study to a small set of test functions. In particular, we introduce implementation tricks to deal with differential operators in large dimensions with structured kernels. Finally, we extend on the core principles beyond our approach to apply them to non-linear spaces of functions, such as the ones parameterized by deep neural networks, through loss-based optimization procedures.

1 INTRODUCTION

Eigen and singular decompositions are ubiquitous in applied mathematics. They can serve as a basis to define good features in machine learning pipelines (Belkin and Niyogi, 2003; Coifman and Lafon, 2006; Balestriero and LeCun, 2022), while a set of good features naturally define pullback distances on the original data. Those features and distances are naturally referred to as "spectral embeddings" and "spectral distances". The latter are thought to provide meaningful geometries on the data, which explain their uses for clustering (Belkin and Niyogi, 2004; Schubert et al., 2018), as well as diffusion models (Chen and Lipman, 2023). In the machine learning community, spectral decompositions are usually derived from the eigen decompositions of different graph Laplacians built on top of the data (Chung, 1997; Zhu et al., 2003; Ham et al., 2004). However, those methods are known to scale poorly with the input dimension (Bengio et al., 2006; Singer, 2006; Hein et al., 2007), although they had applications in many different fields, such as molecular simulation (Glielmo et al., 2021), acoustics (Bianco et al., 2019) or the study of gene interaction (van Dijk et al., 2018).

In this paper, we suggest a different approach to approximate the spectral decompositions of a large class of operators. Our method consists in restricting the study of infinite-dimensional operators on a basis of simple functions, which is usually referred to as Galerkin, Ritz or Raleigh methods (Singer, 1962), if not Bubnov or Petrov (Fluid Dynamics, 2012), depending on the research community. We make the following contributions.

1. We release an algorithm to compute spectral decompositions of a large class of operators that is *statistically and computationally efficient*,[1] and provide experiments to confirm those theoretical findings with numerical analysis.
2. We show that our method prevails over graph-based approaches both statistically and computationally. This opens exciting follow-ups for any spectral-based algorithms, could it be spectral clustering, spectral embeddings, or spectral distances.
3. We show that our method prevails over kernelized algorithms based on the use of representer theorems (Schölkopf et al., 2001; Zhou, 2008). Interestingly, our Galerkin approach can be seen as unifying and generalizing both random features and the Nystrom method. To take a concrete example, for the Laplacian problem considered by Pillaud-Vivien and Bach (2023), their use of the representer theorems leads to implementation requiring $O(n^3d^3)$ flops, while the instantiation of our generic framework provides an implementation in $O(n^2 + n^{3/2}d)$ flops without any loss from a statistical viewpoint, for n the number of samples, and d the sample dimension. This opens exciting follow-ups for any kernel methods dealing with derivatives, such as Hermite regression.
4. Finally, we extract the core principles behind our method in order to consider non-linear spaces of functions through loss-based optimization procedures. We equally discuss how our perspective can model approaches that have led to state-of-the-art results in self-supervised learning.

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

[1] Our Python library can be downloaded through the command line `$ pip install klap`, or built from the source code.

2 SETUP

This section details the setup, motivations, and a running example behind our study.

Data. We assume that n samples $(x_i)_{i\in[n]}$ have been collected and stored as raw vectors $x_i \in \mathcal{X} = \mathbb{R}^d$.[2] The collection process is idealized as underlying a distribution $\rho \in \Delta_{\mathcal{X}}$ which has generated the samples as n independent realizations of the variable $X \sim \rho$.

Goal. Let us consider an operator $\mathcal{L} : L^2(\rho) \to L^2(\rho)$ in a large class of operators. To be precise, we assume that $\mathcal{L}$ has discrete spectrum and defines a bilinear form as an expectation over the data through a known operation $H : L^2(\rho) \times L^2(\rho) \times \mathcal{X} \to \mathbb{R}$ that is supposed to be bilinear in f and g,

$$\mathcal{E}_{\mathcal{L}}(f, g) = \langle f, \mathcal{L}g\rangle_{L^2(\rho)} = \mathbb{E}_X[H(f, g, X)]. \quad (1)$$

For example, $\mathcal{L}$ could be equal to the differential operator defined with the partial derivative ∂_i, in which case $H(f, g, x) = f(x)\partial_i g(x)$. Our goal is to approximate the spectral decomposition of $\mathcal{L}$

$$\mathcal{L} = \sum_{i\in\mathbb{N}^*} \lambda_i f_i \otimes g_i, \quad (2)$$

for $\lambda_i > 0$ the ordered singular values of $\mathcal{L}$ and $f_i, g_i \in L^2(\rho)$ the corresponding left and right singular functions. The decomposition is performed in $L^2(\rho)$, i.e., the (f_i), and respectively the (g_i), are orthonormal in $L^2(\rho)$.

Running example. A prototypical example is provided by

$$H_0(f, g, X) = \langle \nabla f(X), \nabla g(X)\rangle . \quad (3)$$

The resulting operator is the Laplacian $\mathcal{L}_0 = \nabla^*\nabla$ where ∇ is the Euclidean gradient, and the adjoint is taken with respect to the $L^2(\rho)$-geometry.[3] In this case, $\mathcal{L}_0$ is positive self-adjoint, hence $f_i = g_i$.

The operator $\mathcal{L}_0$ has found applications for representation learning, that aims to extract good feature to describe raw data, as the eigenfunctions f_i encode some sort of "modes" on the input space, while the λ_i captures a notion of complexity of those (Bousquet et al., 2003; Cabannes et al., 2023a), similarly to Fourier modes where complexity increases with frequency, see Figure 1.

Interestingly, $\mathcal{L}_0$ also relates to Langevin dynamics, which aims to generate new samples that could have originated from the original data distribution (Dockhorn et al., 2022). In particular, when ρ derives from a potential $V : \mathcal{X} \to \mathbb{R}$, i.e., $\rho(\mathrm{d}x) = \exp(-V(x))\,\mathrm{d}x$ with $\mathrm{d}x$ the Lebesgue measure, under mild assumptions on V, as detailed in Appendix A.1, the operator $\mathcal{L}_0$ is characterized by

$$\mathcal{L}_0 : f \mapsto -\Delta f + \langle \nabla V, \nabla f\rangle .$$

In this setting, (1) is known as the Dirichlet energy, and $\mathcal{L}_0$ as the infinitesimal generator of the Langevin diffusion (Grenander and Miller, 1994; Bakry et al., 2014; Liu and Wang, 2016).[4]

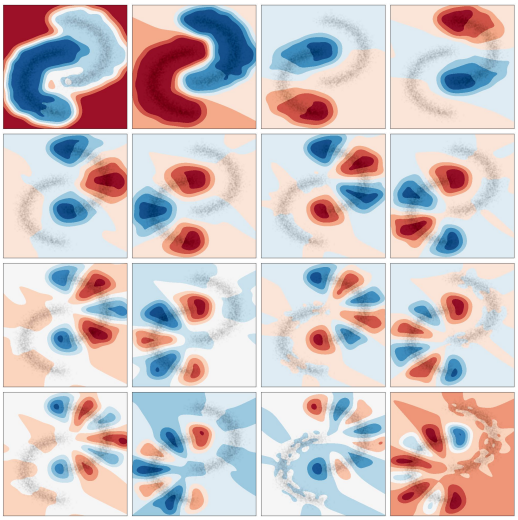

Figure 1: Level lines of the first sixteen learned eigenfunctions of $\mathcal{L}_0$ when the data generates two half-moons with $d = 2$, with Algorithm 3, $n = 10^5$ points, and $p = 200$ Galerkin functions derived from the exponential kernel. See how those eigenfunctions are separated between the two clusters, and how, on each cluster, they identify with Fourier modes (i.e., cosines) when distorting the segment $[0, 1]$ into a half-moon.

3 GALERKIN METHOD

This section introduces our method, and discusses its statistical efficiency. Our method assimilates to the Galerkin method, together with a Monte-Carlo quadrature rule to estimate the scalar product of $L^2(\rho)$ (Chatelin, 1983).

3.1 Algorithm

Rather than working with the infinite-dimensional operator $\mathcal{L}$, we will restrict our study to its action on a finite-dimensional space of functions. To do so, we consider p real-valued functions $(\varphi_i)_{i\in[p]}$ to cast a vector $\alpha \in \mathbb{R}^p$ into a function in $L^2(\rho)$ through the embedding

$$\begin{array}{rccc} F : & \mathbb{R}^p & \to & L^2(\rho); \\ & \alpha & \mapsto & \sum_{i\in[p]} \alpha_i \varphi_i, \end{array} \quad (4)$$

[2] In all the following, we use the notation $[n] = \{1, 2, \ldots, n\}$.

[3] Laplacians are usually defined as negative self-adjoint operators $-\nabla^*\nabla$ as for the usual Laplacian $\Delta = \sum \partial_{ii}^2$ which corresponds to adjunction with respect to $L^2(\mathrm{d}x)$ endowed with the Lebesgue measure. This paper rather uses the graph-based convention where Laplacians are positive self-adjoint operators.

[4] Reciprocally, diffusion models have been proposed as some form of power methods to estimate eigen decomposition by Han et al. (2020).

Algorithm 1: Galerkin method

Data: Data $(x_i) \in \mathcal{X}^n$, function H as per (1).
Choose p test functions $(\varphi_i)_{i\in[p]}$, $(\psi_i)_{i\in[p]}$;
Compute $\hat{L} = (\sum_{k\in[n]} H(\varphi_i, \psi_j, x_k))_{ij} \in \mathbb{R}^{p\times p}$;
Compute $\hat{\Phi} = (\sum_{k\in[n]} \varphi_i(x_k)\varphi_j(x_k))_{ij} \in \mathbb{R}^{p\times p}$;
Compute $\hat{\Psi} = (\sum_{k\in[n]} \psi_i(x_k)\psi_j(x_k))_{ij} \in \mathbb{R}^{p\times p}$;
Solve $(\Lambda, A, B) = \mathrm{GSVD}(\hat{L}, \hat{\Phi}, \hat{\Psi})$ (7);
Set $\hat{\lambda}_i := \Lambda_{ii}$, $\hat{f}_i := \sum_{j\in[p]} A_{ij}\varphi_i$, $\hat{g}_i := \sum B_{ij}\psi_i$;
Result: Estimate $(\hat{\lambda}_i, \hat{f}_i, \hat{g}_i)$ to decompose $\mathcal{L}$ as per (2).

and we search for the singular functions as $f_i = F\alpha_i$ for $\alpha_i \in \mathbb{R}^p$. In full generality, we can distinguish the search space for left and right singular functions, and introduce a second family of real-valued functions $(\psi_i)_{i\in[p]}$ to map $\beta \in \mathbb{R}^p$ to a function in $L^2(\rho)$ through the embedding G defined from the (ψ_i). Restricted to those parametric functions, the operator $\mathcal{L}$ assimilates to the *matrix* $L \in \mathbb{R}^{p\times p}$, defined as

$$L_{ij} := (F^*\mathcal{L}G)_{ij} = (\mathbb{E}_X[H(\varphi_i, \psi_j, X)])_{ij}, \qquad (5)$$

where the adjunction is taken with respect to $L^2(\rho)$ and $\mathbb{R}^p$, and the last equation is proven in Appendix A.2.

The search for the decomposition (2) reduces to the search of the decomposition,

$$L = \sum_{i\in\mathbb{N}^*} \lambda_i F^* f_i g_i^* G \approx \sum_{i\in[t]} \lambda_i F^* F\alpha_i\beta_i^* G^* G,$$

where the last equation is due to our search for f_i and g_i under the form $F\alpha_i$ and $G\beta_i$, together with a threshold $t \in \mathbb{N}$. Introducing the $t \times p$ matrices $A = (\alpha_i)_{i\in[t]}$, $B = (\beta_i)$ and $\Lambda = \mathrm{diag}(\lambda_i) \in \mathbb{R}^{t\times t}$, the last equation is rewritten as

$$L \approx F^* F A^\top \Lambda B G^* G.$$

In addition to the above decomposition, we should specify orthogonality constraints,

$$\langle F\alpha_i, F\alpha_j\rangle_{L^2(\rho)} = \alpha_i F^* F\alpha_j = \alpha_i \Phi \alpha_j = \delta_{ij},$$

where $\Phi \in \mathbb{R}^{p\times p}$ is defined as (see Appendix A.2)

$$\Phi_{ij} = (F^*F)_{ij} = (\mathbb{E}[\varphi_i(X)\varphi_j(X)])_{ij}. \qquad (6)$$

Considering empirical averages in the definition of L in (5), Φ and Ψ in (6) leads to Algorithm 1, where the generalized singular value decomposition (GSVD) of $(\hat{L}, \hat{\Phi}, \hat{\Psi})$ reads

$$A\hat{L}B^\top = \Lambda, \qquad A\hat{\Phi}A^\top = B\hat{\Psi}B^\top = I, \qquad (7)$$

where $A, B, \Lambda \in \mathbb{R}^{p\times p}$ and Λ is diagonal with positive entries. This decomposition exists as soon as $t = p$ and $\hat{\Phi}$ and $\hat{\Psi}$ are invertible, as shown in Appendix A.3.

Computational complexity. Our method consists in building and storing the $p\times p$ matrices $\hat{L}$, $\hat{\Phi}$ and $\hat{\Psi}$, before solving the associated generalized singular value decompositions. The building of the matrix $\hat{L}$ scales in $O(np^2c_H)$ flops, where c_H is the cost of evaluating $H(\varphi_i, \varphi_j, x_k)$, while the decomposition scales in $O(p^3)$. As a consequence, the overall number of flops is $O(np^2c_H + p^3)$. In terms of memory, we only need $O(p^2 + b_H)$ bits of memory where b_H is the memory needed to evaluate a single $H(\varphi_i, \psi_j, x_k)$.

3.2 Statistical Efficiency

Our algorithm can be seen through two actions. First, the operator $\mathcal{L}$ in (1) is restricted to its action on $\mathrm{im}\, F$ and $\mathrm{im}\, G$ with the introduction of the operator L in (5). Second, because $\mathrm{im}\, L$ and $\mathrm{im}\, G$ are finite-dimensional, the operator L assimilates to a matrix. This matrix is defined from the distribution ρ but is approximated with empirical data, leading to $\hat{L}$, which will concentrate around L as the number of data grows. Those facts are captured formally by Theorem 1, proven in Appendix A.4. It introduces Π_F the orthogonal projector on $\mathrm{im}\, F$ in $L^2(\rho)$. It focuses on the reconstruction of the inverse $\mathcal{L}^{-1}$ in operator norm, based on the estimation suggested by Algorithm 1. Controlling the operator norm allows the reconstruction of both the spectral values and functions (e.g., Weyl, 1912). Our choice to focus on $\mathcal{L}^{-1}$ is due to the fact that when $\mathcal{L}$ is a differential operator, $\mathcal{L}$ is usually not bounded, but its inverse is compact (Kondrachov, 1945).

Theorem 1. *Assume that $H(\varphi_i, \psi_j, x)$ is bounded by H_∞ independently of (i, j, x), and that L is invertible. For any $\delta > 0$, and $n > 3\max(1, p^2H_\infty^2\|L^{-1}\|^{-2})\log(2p/\delta)$, the following holds true with probability at least $1-\delta$ (the randomness coming from the data),*

$$\begin{aligned}\left\|\mathcal{L}^{-1} - G\hat{L}^{-1}F^*\right\| &\leq 3\|\Psi^{1/2}L^{-1}\|\|L^{-1}\Phi^{1/2}\|\sqrt{\frac{8p^2H_\infty^2}{3n}\log(\frac{2p}{\delta})}. \\ &\quad + \left\|(I-\Pi_G)\mathcal{L}^{-1}\right\| + \left\|\mathcal{L}^{-1}(I-\Pi_F)\right\|, \qquad (8)\end{aligned}$$

where $\|\cdot\|$ is the operator norm.

The first term in Theorem 1 can be estimated empirically, it relates to the variance of our estimate. The second and third term relates to the bias of our estimator, and cannot be known without specific assumptions on the spectral decomposition of the operator $\mathcal{L}$.

In the following, we will consider that the features φ_i and ψ_i were actually obtained as independent realizations of a random variable φ and ψ. In those settings, we introduce the following operators in $L^2(\rho)$

$$\Sigma = \mathbb{E}_\varphi[\varphi\varphi^*], \qquad \Xi = \mathbb{E}_\psi[\psi\psi^*],$$

where the adjunction is understood in $L^2(\rho)$. The projection on F naturally becomes the projection on $\mathrm{im}\,\Sigma^{1/2}$. The following theorem, whose proof is provided in Appendix A.5, refines Theorem 1 in this particular situation.

Theorem 2. *Let $(\psi_i)_{i\in[p]}$ be p independent realizations of a random function ψ. Let Π_p denotes the projection on the span of the first p spectral functions of Ξ, and $\mathcal{L}_p = \sum_{i\in[p]} \lambda_i f_i g_i^*$. Assume that $\|\psi\|_{L^2(\rho)} \leq M$ almost surely. In the setting of Theorem 1, for any δ, with probability $1-\delta$ (the randomness being understood with respect to the random functions), when $p > 3\log(2/\|\Sigma\|\,\delta)$,*

$$\left\|(I-\Pi_G)\mathcal{L}^{-1}\right\| \leq \lambda_{p+1}^{-1} + \left\|(I-\Pi_p)\mathcal{L}_p^{-1}\right\| + \left\|\Pi_p\Sigma^{-1}\mathcal{L}_p^{-1}\right\| \sqrt{\frac{8M^2}{3p}\log(2p/\delta)}. \quad (9)$$

In Theorem 2, λ_{p+1}^{-1} captures how fast the spectrum of $\mathcal{L}^{-1}$ vanishes, $\left\|(I-\Pi_p)\mathcal{L}_p^{-1}\right\|$ depends on how well-specified our model is to retrieve the first p spectral functions of $\mathcal{L}$, and $\left\|\Pi_p\Xi^{-1}\mathcal{L}_p^{-1}\right\|$ relates to the variance of our estimator. In order to understand the operator norms appearing in the previous theorems, notice that when f_i and g_i are singular functions of Σ and Ξ, we have

$$\|\Sigma^{-1/2}\mathcal{L}^{-1}\Xi^{-1/2}\| = \sup_{i\in\mathbb{N}^*} \lambda_i^{-1}\|\Sigma^{-1/2}f_i\| \cdot \|\Xi^{-1/2}g_i\|.$$

In other terms, ensuring this norm to be bounded consists in enforcing that the complexity to reconstruct f_i from φ (captured with $\|\Sigma^{-1/2}f_i\|$) and g_i from ψ does not grow too fast compared to the vanishing rates of the sequence (λ_i). In particular, when (λ_i) decreases slowly, we need to ensure $\|\Sigma^{-1/2}f_i\|$ to be small for many indices in order to guarantee good reconstruction properties, while when it decreases fast, only approximating well the first few spectral functions guarantee a similar overall reconstruction error. Those considerations are illustrated with a concrete example in Section 5.

4 DIFFERENTIAL OPERATORS, INVARIANT KERNELS

This section discusses a large class of operators that take the form of equation (1), and natural spaces of functions to instantiate our algorithm.

Differential operators. A large class of interest lies in the operators defined through the bilinear form (1) with

$$H_c(f,g,x) = \sum_{\alpha,\beta\in\mathbb{N}^d} c_{\alpha,\beta}\partial_\alpha f(x)\partial_\beta g(x), \quad (10)$$

where for $\alpha = (\alpha_1,\ldots,\alpha_d) \in \mathbb{N}^d$, $\partial_\alpha f$ denotes the partial derivatives of f where the i-th coordinate is derived α_i-times, and $c = (c_{\alpha,\beta})$ is a sequence in $\mathbb{R}$ indexed by $\mathbb{N}^d \times \mathbb{N}^d$ with a finite number of non zero elements. This class encompasses the operator $\mathcal{L}_0$ in (3), defined through

$$H_0(f,g,X) = \sum_{i\in[d]} \partial_i f(x)\partial_i g(x),$$

where ∂_i denote the partial derivative with respect to the i-th variables, formally $\partial_i = \partial_{e_i}$ with e_i the i-th vector of the canonical basis in $\mathbb{R}^d$.

Reproducing kernel Hilbert spaces. In many cases, the spectral functions of an operator $\mathcal{L}$ are known to belong to a restricted set of functions $\mathcal{H}$. For example, spectral functions of most differential operators are expected to belong to the Sobolev spaces H^s for many s (Shubin, 1987). Moreover, the set $\mathcal{H}$ can often be endowed with a Hilbertian structure that makes evaluation map $k_x^* : f \mapsto f(x)$ bounded. In this case, k_x^* can be represented with $k_x \in \mathcal{H}$ as $k_x^*(f) = \langle f, k_x\rangle_{\mathcal{H}}$ which is the basis of reproducing kernel (Schölkopf and Smola, 2001; Berlinet and Thomas-Agnan, 2011). Typical examples are provided by polynomials of degree less than s, by the class of Sobolev functions $H^{(d+1)/2}$, or by some subspace of analytical functions, for which one can consider respectively

$$k_x(y) = q(x^\top y), \quad \text{with} \quad q(t) = (1+t)^s, \quad (11)$$

and with $\beta = 1$ and $\beta = 2$ respectively,

$$k_x(y) = q(\|x-y\|), \quad \text{with} \quad q(t) = \exp(-t^\beta). \quad (12)$$

The space $\mathcal{H}$ equates the closure with respect to its inner product of the span of the k_x for $x \in \mathcal{X}$.

Low-dimensional approximation. Reproducing kernel Hilbert spaces are usually infinite-dimensional. However, when dealing with a finite number of data, they can be restricted on a finite-dimensional space as a consequence of representer theorems (Schölkopf et al., 2001). More generally, most of the action on $\mathcal{H}$ can be restricted to the span of a few number of elements k_{x_i} for $x_i \sim \rho$ (Williams and Seeger, 2000), or to the span of few "random features" (Rahimi and Recht, 2007), which leads to strong computational gain for a low statistical cost (Rudi et al., 2015; Rudi and Rosasco, 2017). Following the first idea, we set

$$\varphi_i = \psi_i = k_{x_i}, \quad (13)$$

for $(x_i)_{i\in[p]}$ subsampled from the data $(x_i)_{i\in[n]}$, i.e., $p < n$. The difference between the projections of spectral functions on $\operatorname{im} F = \operatorname{Span}\{k_{x_i}\}$ and their projections on $\mathcal{H}$ is naturally characterized from the difference between the covariance $\Sigma = \mathbb{E}[k_X k_X^*]$ and its empirical estimate $\hat{\Sigma}$ from the points $(x_i)_{i\in[p]}$, which, when the former has a rapidly decaying spectrum, can vanish quite fast as p increases.

Implementation. When considering the objective (10) with the space defined in (13), $\hat{L}$ becomes

$$\hat{L}_{ij} = \sum_{k,\alpha,\beta} c_{\alpha,\beta}\partial_\alpha k_{x_i}(x_k)\partial_\beta k_{x_j}(x_k).$$

Often, the structure of this matrix allows to reduce, at the cost of an increase in memory space, the number of flops needed to build it compared to a naive implementation. The next section illustrates this fact through Proposition 1 for the specific case where $\mathcal{L} = \mathcal{L}_0$.

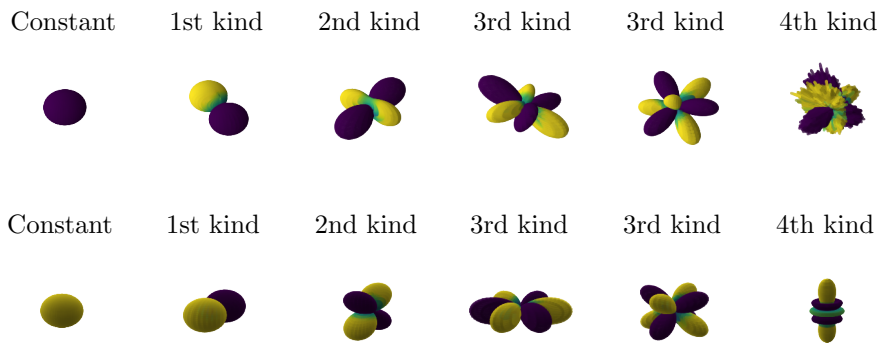

Figure 2: Learning spherical harmonics with polynomials of degree three (with $k_x(y) = (1 + x^\top y)^3$ which corresponds to features maps that concatenates all the multivariate monomials of degree smaller or equal to $s = 3$). Because we consider ρ uniform on the sphere, the operator $\mathcal{L}_0$ is diagonalized by spherical harmonics, which are polynomials of increasing degrees. The polynomial kernel of degree D allows to learn all harmonics of s-th kind for s smaller or equal to D (the ones of higher kind are polynomials of higher degree that can not be reconstructed with polynomials of degree D as illustrated with the fourth kind on the figure). Some of the learned eigenfunctions are represented on the top row, while some ground truths are represented on the bottom row. Our method learned perfectly valid harmonics, although, for eigenvalues that are repeated, it does not learn the canonical ones, but any basis of the different eigenspaces (which can be observed with the harmonics of the second kind in the figure).

5 THE LAPLACIAN EXAMPLE

This section details previous considerations for the specific example where $\mathcal{L} = \mathcal{L}_0$ in (3).

Specificities. First of all, $\mathcal{L}_0$ is self-adjoint positive, inheriting its symmetry and positiveness from H_0 in (3). Moreover, under mild assumptions on ρ, the Rellich–Kondrachov embedding theorem holds true, which implies the compactness of $\mathcal{L}^{-1}$ (Kondrachov, 1945). This is useful to apply the spectral theorem and consider the countable eigenvalue decomposition of $\mathcal{L}_0$. Finally, because $\mathcal{L}_0$ is a diffusion operator, in some cases of interest, its eigenfunctions are polynomials (Bakry et al., 2021), e.g., when $\rho = \mathcal{N}(0, I)$ or when ρ is uniform on the sphere $\mathbb{S}^{d-1}$.

Statistical efficiency. We now turn to the statistical efficiency of our estimator. The next theorem, proven in Appendix A.6, shows how our algorithm succeeds in leveraging the smoothness of spectral functions to guarantee fast rates of convergence.

Theorem 3. *Assume that $\|x\| \leq M$ almost surely, i.e., ρ has compact support, and that $\mathcal{L}_0$ is diagonalized by polynomials of increasing order. For $s \in \mathbb{N}$, consider the kernel $k_x(y) = (.5 + x^\top y/2M^2)^s$ and the search for eigenfunctions with* (13)*. For any $\delta \in (0, 1)$, there exists a constant c_δ such that for n large enough, $p = n^{1/2}$ and $s = s_n$ (set so $d \log(\lambda_s) + \log(s) = \log(n)/2$), with probability $1 - \delta$, Algorithm 1 with $\varphi_i = \psi_i = k_{x_i}$ guarantees*

$$\left\| \mathcal{L}^{-1} - G\hat{L}^{-1}F^* \right\| \leq c_\delta n^{-d/2(d+1)}.$$

This theorem should be compared with the study of Pillaud-Vivien and Bach (2023) that gives rates in $O(n^{-1/4})$ with an algorithm based on a representer theorem that leads to an implementation in $O(n^3d^3)$ flops and $O(n^2d^2)$ memory bits. In contrast, we can ensure better rates for only $O(n^2 + n^{3/2}d)$ flops and $O(n^{3/2})$ bits.[5] It should equally be compared with graph-Laplacian approaches whose error scales in $O(n^{-1/d})$ for at least $O(n^2p + n^3)$ flops and $O(n^2)$ bits to build and get the eigen decomposition of the graph-Laplacian matrix (which we actually reduce to $O(n^2p + np^2 + npd)$ flops and $O(n^2 + np)$ bits in Appendix B.1). Although the result of Theorem 3 might seem to break the curse of dimensionality, it should be noted that it could hide constant that grows exponentially fast with respect to the dimension, which may limit the application of our method to large dimensional problems where it is impossible to get a good estimation of ρ without a large number of data (Cabannes and Vigogna, 2023).

Implementation. We conclude this section by showing that our method can be implemented with $O(np^2 + npd)$ flops and $O(p^2 + nd)$ memory bits.

Proposition 1. *Assume that $\mathcal{X}$ is endowed with a scalar product. Given a kernel $k_x(y) = q(\|x - y\|)$ defined from $q : \mathbb{R} \to \mathbb{R}$, for $x, y, z \in \mathcal{X}$, we have*

$$\begin{aligned} H_0(k_y, k_z, x) &= \langle \nabla k_y(x), \nabla k_z(x) \rangle \\ &= \frac{q'(\|x - y\|)}{\|x - y\|} \frac{q'(\|x - z\|)}{\|x - z\|} (x - y)^\top (x - z). \end{aligned} \quad (14)$$

Similarly for dot-product kernel $k_x(y) = q(x^\top y)$,

$$H_0(k_y, k_z, x) = q'(x^\top y) q'(x^\top z)\, y^\top z. \quad (15)$$

Proof. The proof follows from the application of the chain rule in the calculation of $\langle \nabla_x k_y(x), \nabla_x k_z(x) \rangle$. □

Our implementation computes: *(i)* $X = (x_i^\top x_k)_{ik} \in \mathbb{R}^{p \times n}$ with $O(npd)$ flops and $O(np)$ bits; *(ii)* $q(X)$ and $q'(X)$,

[5] To see this, plug $p = n^{1/2}$ into the complexity bounds in $O(np^2 + npd)$ and $O(np)$ of the next paragraph.

where the operator is understood elements wise, with $O(np)$ flops and $O(np)$ bits; *(iii)* Ψ and L from $q(X)$, $q'(X)$ and X with $O(np^2)$ flops and $O(np)$ bits; *(iv)* the generalized eigen decomposition (GEVD) associated with (L, Ψ) with $O(p^3)$ flops in $O(p^2)$ bits.[6] Adding the steps leads to a total of $O(npd + np^2 + p^3)$ flops and $O(np)$ bits.

Algorithm 2: $\mathcal{L}_0$ estimate with dot-product kernel

Data: Data $(x_i) \in \mathcal{X}^n$, kernel $k_x(y) = q(x^\top y)$.
Compute $X = (x_i^\top x_j) \in \mathbb{R}^{p\times n}$ with $p \leq n$;
Compute $\Psi = q(X)q(X)^\top \in \mathbb{R}^{p\times p}$ elementwise;
Compute $L = (q'(X)q'(X)^\top)$;
Update $L_{ij} \leftarrow X_{ij}L_{ij}$ for all $i, j \in [p]$;
Solve $(\hat\lambda_i, (\alpha_{ij})_{j\in[p]})_{i\in[p]} \leftarrow \mathrm{GEVD}(L, \Psi)$;
Set $\hat f_i(x) := \sum_{j\in[p]} \alpha_{ij} k_{x_j}(x)$.
Result: Estimate $(\hat\lambda_i, \hat f_i)$ of the decomposition of $\mathcal{L}_0$.

Algorithm 3: $\mathcal{L}_0$ estimate with distance kernel

Data: Data $(x_i) \in \mathcal{X}^n$, kernel $k_x(y) = q(\|x - y\|)$.
Compute $X = (x_i^\top x_j), \in \mathbb{R}^{p\times n}$, $D = (x_i^\top x_i) \in \mathbb{R}^n$;
Deduce $N = (\|x_i - x_j\|) \in \mathbb{R}^{p\times n}$ and $T = q'(N)/N$;
Initialize $L = 0 \in \mathbb{R}^{p\times p}$; $\Psi = q(N)q(N)^\top \in \mathbb{R}^{p\times p}$;
for $k \in [n]$ **do**
  Set $\gamma_{ij}^{(k)} := (D_k - X_{ik} - X_{jk} + X_{ij})$;
  Update $L_{ij} \leftarrow L_{ij} + \gamma_{ij}^{(k)} T_{ik} T_{jk}$;
Solve $(\lambda_i, (\alpha_{ij})_{j\in[p]})_{i\in[p]} \leftarrow \mathrm{GEVD}(L, \Psi)$;
Set $f_i(x) := \sum_{j\in[p]} \alpha_{ij} k_{x_j}(x)$.
Result: Estimate (λ_i, f_i) of the decomposition of $\mathcal{L}$.

6 RELATED APPROACHES

6.1 Graph Laplacians

Graph Laplacians are the classical way to estimate spectral decompositions in machine learning (Zhu et al., 2003; Belkin and Niyogi, 2003; Zhu et al., 2021; Zhu and Koniusz, 2022). Although there exist many variants, those methods mainly consist in approximating the Laplacian operator $\mathcal{L}_0$ with finite differences. For $f : \mathcal{X} \to \mathbb{R}$

$$\mathbb{E}[\|\nabla f(X)\|^2] \approx \sum_{i,j\in[n]} w_{ij}(f(x_i) - f(x_j))^2, \qquad (16)$$

where the w_{ij} are a set of weights usually taken as $(w_{ij}) = D^{-1/2}\tilde W D^{-1/2}$ where $\tilde w_{ij} = \exp(-\alpha\|x_i - x_j\|^2)$ with α a scale parameter, and D is the diagonal matrix with $D_{ii} = \sum_{j\in[n]} \tilde w_{ij}$.

Graph Laplacians differ from our approaches in several aspects. On the positive side, they could present computational advantages, when the evaluation of H_0 in (3) is too costly. Moreover, by tuning the weighting scheme w, graph Laplacians can estimate different Laplacians, such as the Laplace-Beltrami operator associated with the data manifold even if the data are non uniform on this manifold (Coifman and Lafon, 2006; Hein et al., 2007), while this operator usually can not be written under the form needed to apply our framework from (1). However, approximating a differential operator with finite differences is known to be statistically inefficient in high dimension (Bengio et al., 2006; Singer, 2006; Hein et al., 2007), as it will not easily adapt to the smoothness of the targeted operator. Finally, graph-Laplacians are used to approximate Laplacian operators, while our method is more generic.

6.2 Methods based on Loss Optimization

In the era of deep learning, one of the main challenges of machine learning pipelines is to design a principled loss, before finding an architecture and an optimizer which can practically minimize this loss when choosing a good set of hyperparameters. This section explores the core principles beyond our approach, and details the possibility to learn spectral decompositions with any models, including deep neural networks. To ease the discussion, we assume that $\mathcal{L}$ is symmetric and thus that $f_i = g_i$ in the following.

In this paper, we started from an objective term $\mathcal{E}_\mathcal{L} : L^2(\rho) \to \mathbb{R}$, whose minimization of $\sum_{i\in[p]} \mathcal{E}_\mathcal{L}(f_i)$ under the constraints that the (f_i) are orthogonal in $L^2(\rho)$ retrieves the first eigenspace of $\mathcal{L}$, viz., the span of the f_i equates the one of the first eigenfunctions $(f_i^*)_{i\in[p]}$ of $\mathcal{L}$ (2). This property holds true from any unitary norm (Mirsky, 1960), leading to many losses that could be used to train a neural network.

Second, we needed a way to enforce orthogonality between the different f_i's, which led us to consider generalized singular value decomposition. When the f_i's are learned with optimization methods, one might naively add a regularizer to the objective that reads, with $f : \mathcal{X} \to \mathbb{R}^p$ whose coordinates are the f_i's,

$$\begin{aligned}\mathcal{R}(f) &= \left\|\mathbb{E}[f(X)f(X)^\top] - I\right\|^2 \\ &= \mathbb{E}[(f(X)^\top f(X'))^2] - 2\mathbb{E}[\|f(X)\|^2] + p, \quad (17)\end{aligned}$$

where X' is another independent realization of X, the last equation being useful in order to get unbiased estimates from small batches of this objective. Minimizing the resulting loss $\mathcal{E}_\mathcal{L} + \mathcal{R}$ allows to retrieve the spectral decomposition of $\mathcal{L}$, since

$$\begin{aligned}\operatorname*{arg\,min}_{f:\mathcal{X}\to\mathbb{R}^p} 2\sum_{j\in[p]} \mu_i^{-1}\mathcal{E}_\mathcal{L}(f_j) + \mathcal{R}(f) \\ = \left(\sqrt{\left(1 - \frac{\lambda_i}{\mu_i}\right)_+} f_i\right)_{i\in[p]}. \quad (18)\end{aligned}$$

A proof of this statement can be found in Zhang et al. (2022) (see also Johnson et al., 2023; Cabannes et al., 2023b).

[6] In practice, one might regularize the system as $(L + \varepsilon I, \Psi)$ or $(L, \Psi + \varepsilon I)$ for a small regularizer $\varepsilon > 0$ to avoid diverging solution $\alpha_i = +\infty$ due to inversion instability, especially if the eigenfunctions of $\mathcal{L}$ do not belong to our parametric models.

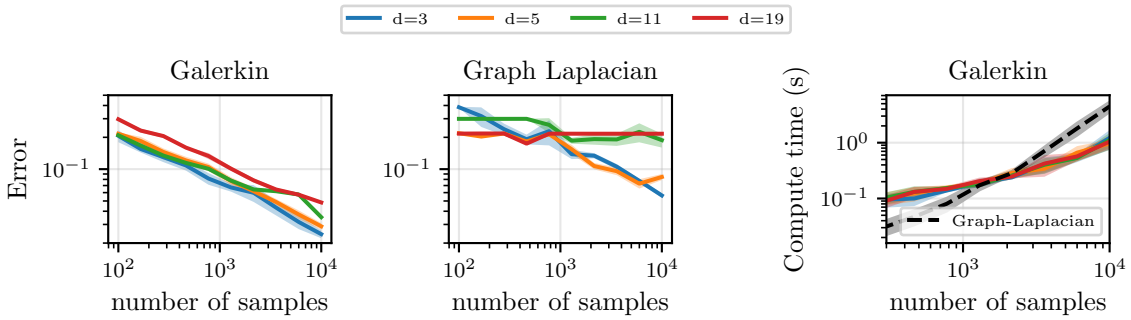

Figure 3: (Left) Testing error (21) when learning the first 25 "spherical harmonics" eigenvalues as a function of the number of samples n in different dimension d with Galerkin method. (Middle) Same figure with graph-Laplacian. The error is averaged over 100 runs, with standard deviations shown in solid color, and we pick the best result over three kernels with five different parameters each with five different values for p (best of 75 for Galerkin), as well as six different scales for weighting in graph-Laplacian (best of 450 for graph-Laplacian). (Right) Computation time for Galerkin method with polynomial kernel of degree three and $p = 177$. Experimental setups and reproducibility specifications are detailed in Appendix B.

6.3 Modeling of Self-Supervised Learning

Many representation learning methods can be modeled as estimating the first spectral functions of some operators which are compatible with our framework. In the pre-deep learning area, the operator $\mathcal{L}_0$ in (3) was the operator of choice to extract "spectral embeddings". Several self-supervised learning algorithms can be seen as using variations of this operator. First, Simard et al. (1991) suggested learning features that are invariant to small perturbations locally by working with what we would call today "Jacobian vector products". It fell within our framework with

$$\mathcal{E}_{\mathcal{L}_{\text{TP}}}(f) = \mathbb{E}_X \mathbb{E}_U \left[\langle \nabla f(x), U \rangle^2 \,\middle|\, X \right], \qquad (19)$$

where the distribution $(U \mid X = x)$ specifies the direction of invariance to be enforced at the point x. When U is uniform on the sphere, we retrieve $\mathcal{L}_0$, although this estimator will suffer from high variance as detailed in Appendix A.7.

More recently, invariance has been enforced by looking at finite differences between different data augmentations $\xi, \xi' \in \mathbb{R}^d$ of the same original image x (e.g., Chen et al., 2020). With our formalism, this would be written

$$\mathcal{E}_{\mathcal{L}_{\text{SSL}}}(f) = \mathbb{E}_X \left[\mathbb{E}_{\xi,\xi'} \left[\|f(\xi) - f(\xi')\|^2 \,\middle|\, X \right] \right], \qquad (20)$$

if using the square loss to enforce equality between the representations $f(\xi)$ and $f(\xi')$ of the two versions ξ and ξ' obtained from X after data augmentation.[7] In addition to the energy term $\sum_i \mathcal{E}_{\mathcal{L}}(f_i)$, self-supervised losses enforce the features (f_i) to differ from one another, either through the use of contrastive pairs (Chen et al., 2020) that can be understood through the second equation of (17) (HaoChen et al., 2021), or through the use of a "whitening regularizer" (Ermolov et al., 2021; Zbontar et al., 2021; Bardes et al., 2022) that assimilates to the first equation of (17) (Balestriero and LeCun, 2022). In other terms, those approaches could be modeled through the loss (18) and understood as learning spectral embeddings. Similarly, vision-language models with contrastive language-image pre-training (CLIP) could be modeled with an asymmetric operator defined as per (1).

[7] In the literature, $\mathcal{L}$ is often rewritten as the integral operator associated with the kernel $p(\xi, \xi')/\sqrt{p(\xi)p(\xi')}$ assuming that $p(\xi) = p(x)$, with p denoting the different densities against the Lebesgue measure (assumed to exist) (HaoChen et al., 2021; Lee et al., 2021; Deng et al., 2022).

7 EXPERIMENTS

7.1 Spherical Harmonics

In order to check the validity of our methods, we experiment in settings where the ground truth is known. To this end, let us consider the sphere $\mathcal{X} = \mathcal{S}^{d-1}$ in $\mathbb{R}^d$ with the uniform distribution. The operator $\mathcal{L}_0$ in (3) identifies to the square of the orbital angular momentum (Condon and Shortley, 1935; Frye and Efthimiou, 2014), whose eigenfunctions are known to be the spherical harmonics. Those are polynomials of increasing order. In particular, there are $N(d, s)$ independent polynomials of degree s, each of them associated with the eigenvalues $\lambda_p = \mu_s$ where, as proven by Frye and Efthimiou (2014, Theorem 4.4 and Proposition 4.5)

$$\mu_s = s(s + d - 2), \; N(d, s) = \frac{2s + d - 2}{s} \binom{s + d - 3}{s - 1}.$$

Figure 2 illustrates how our Galerkin approach enables the learning of spherical harmonics. In order to evaluate the quality of our method, because the operator norm in $L^2(\rho)$ cannot be computed empirically, we use the surrogate metric

$$\mathcal{E}_S(\hat{\lambda}) \propto \sum_{i \in [k]} |\lambda_i^{-1} - \hat{\lambda}_i^{-1}|, \qquad \mathcal{E}_S(0) = 1, \qquad (21)$$

for $k = 25$, which is bounded by $k\|\mathcal{L}^{-1} - G^* L^{-1} F\|$ when the retrieved eigenfunctions $(\hat{f}_i)$ are orthogonal in $L^2(\rho)$ as a consequence of Weyl's theorem (Weyl, 1912).

The left of Figure 3 shows how the eigenvalues retrieved by our method lead to an error $\mathcal{E}_S \approx cn^{1/2}$, since we go from $\mathcal{E}_S \approx 2$ to $\mathcal{E}_S \approx 0.2$ when going from $n = 10^3$ to $n = 10^5$, independently of the dimension (although the constant c increases as illustrated by the offset of the red curve). In contrast, as showcased by the middle of Figure 3, graph Laplacians suffer from the curse of dimensionality. The right of Figure 3 equally shows how the computation time scales more or less linearly with n when p is given and does not depend much on dimension.[8] A more thorough discussion on our experimental setup, on our removal of confounders, and on the effect of the different parameters at play is provided in Appendix B.

7.2 Hermite Regression

This paper presents a novel method which might find several applications (e.g., Cabannes et al., 2021, for semi-supervised learning). For example, it opens the path for Hermite interpolation (Hermite, 1877), where given a set of n data points (x_i), one tries to learn a function f that interpolates both $f(x_i) = y_i$ and $\nabla f(x_i) = t_i$ for some known scalar values (y_i) and vectors (t_i) in $\mathbb{R}^d$. Hermite interpolation is usually approached with the least squares problem,

$$\operatorname*{arg\,min}_{f \in \mathcal{F}} \sum_{i \in [n]} \left\{ (f(x_i) - y_i)^2 + \|\nabla f(x_i) - t_i\|^2 \right\},$$

with $\mathcal{F}$ some search space of functions from $\mathbb{R}^d$ to $\mathbb{R}$. Depending on $\mathcal{F}$, the argument of the minimum might not interpolate the data and their derivatives, in which case the method could be called Hermite regression. Notice that the loss we are considering can be rewritten as

$$\|f(x) - y\|^2 + \|\nabla f(x) - t\|^2 = \|Df - z\|^2,$$

where, with (t_i) now denoting the coordinates of $t \in \mathbb{R}^d$,

$$Df = (f, \partial_1 f, \dots \partial_d f)^\top, \quad z = (y, t_1, \dots, t_d) \in \mathbb{R}^{d+1}.$$

In other terms, Hermite regression is a instance of the more generic type of linear regression problems, i.e.,

$$\operatorname*{arg\,min}_{f : \mathcal{X} \to \mathbb{R}} \mathbb{E}_{(X,Z)}[\|Df(X) - Z\|^2] = (D^*D)^{-1} D^* f_z,$$

with $f_z(x) = \mathbb{E}[Z \,|\, X = x] \in \mathbb{R}^d$ and the adjunction understood in $L^2(\mathbb{R}^d, \mathbb{R}^d, \rho)$ with the ℓ^2-product topology.

Classical approaches based on representer theorems typically require $O(n^3 d^3)$ flops (Zhou, 2008). In contrast, for Hermite regression, we have already seen how to build an approximation of the operator $D^*D = \mathcal{L}_0 + I$ with

[8] The left part of the plot actually shows a better scaling then linear which can be explained by the fact that we are plotting $t_n \simeq c_1 np^2 + c_2 pnd + c_3 p^3$ with c_3, due matrix inversion, expected to be much bigger than c_1 and c_2, due to matrix multiplication.

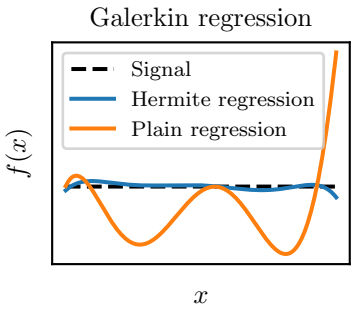

Figure 4: Comparison between plain regression and "Hermite regression" with the Gaussian kernel, $n = 1000$ and $p = 100$ when learning a constant function without noise (a task known to be hard for the Gaussian kernel).

$O(np^2 + npd)$ flops. An approximation of the second term $D^* f_z$ is easier to build, and a naive implementation leads to $O(npd)$ flops. The resulting algorithms, Algorithm 4 and 5 in Appendix B.4, cut down cost to $O(np^2 + npd)$ flops, matching kernel ridge regression implementations that can "handle billions of points effectively" (Meanti et al., 2020). In other terms, we cut down the computational bottleneck associated with the usage of derivatives in kernel methods.

8 CONCLUSION

This paper introduced an algorithm that can be seen as an instance of the Galerkin method to compute the spectral decomposition of a large class of operators. It showcased its usefulness theoretically through a series of approximation guarantees. It was put in perspective with graph-Laplacians, which can be seen as estimating differential operators with finite differences, and suffer from the curse of dimensionality. Those statistical considerations were validated empirically. We later detailed efficient implementations of our approach with structured reproducing kernels, which we have packaged into a Python library to be used off-the-shelf. Those efficient implementations break down computational bottlenecks arising when dealing with derivatives in kernel methods based on representer theorems. In particular, we show how one may perform Hermite regression with $O(np^2 + npd)$ flops, with p chosen by the user, instead of a naive implementation that would scale in $O(n^3 d^3)$.

Finally, we discussed the core principle beyond our approach to design losses whose optimization enables learning the spectral decomposition of linear operators with non-linear spaces of functions, e.g., with deep neural networks. Those losses were designed to be convex on the cone of positive matrices $(f(X)f(X)^\top)$, although when models are not convex, training dynamics might exhibit robustness and stability issues, requiring proper hyperparameters tuning to induce behaviors of interest. We equally extended on how our setting models recent approaches to representation learning, at least when losses are approximated with squared distances, unveiling abstract linear operators beyond them.

Acknowledgement. The authors would like to thank Loucas Pillaud-Vivien and Ricky Chen for fruitful discussions.

References

Dominique Bakry, Ivan Gentil, and Michel Ledoux. *Analysis and Geometry of Markov Diffusion Operators*. Springer, 2014.

Dominique Bakry, Stepan Orevkov, and Marguerite Zani. Orthogonal polynomials and diffusion operators. *Annales de la Faculté des Sciences de Toulouse*, 30(5):985–1073, 2021.

Randall Balestriero and Yann LeCun. Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods. In *NeurIPS*, 2022.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR*, 2022.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NeurIPS*, 2004.

Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. Label propagation and quadratic criterion. In *Semi-Supervised Learning*. MIT Press, 2006.

Alain Berlinet and Christine Thomas-Agnan. *Reproducing Kernel Hilbert spaces in Probability and Statistics*. Springer Science & Business Media, 2011.

Michael Bianco, Peter Gerstoft, James Traer, Emma Ozanich, Marie Roch, Sharon Gannot, and Charles-Alban Deledalle. Machine learning in acoustics: Theory and applications featured. *The Journal of the Acoustical Society of America*, 146(5):3590–3628, 2019.

Olivier Bousquet, Olivier Chapelle, and Matthias Hein. Measure based regularization. In *NeurIPS*, 2003.

Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.

Vivien Cabannes and Stefano Vigogna. How many samples are needed to leverage smoothness? In *NeurIPS*, 2023.

Vivien Cabannes, Loucas Pillaud-Vivien, Francis Bach, and Alessandro Rudi. Overcoming the curse of dimensionality with Laplacian regularization in semi-supervised learning. In *NeurIPS*, 2021.

Vivien Cabannes, Alberto Bietti, and Randall Balestriero. On minimal variations for unsupervised representation learning. In *ICASSP*, 2023a.

Vivien Cabannes, Bobak Kiani, Randall Balestriero, Yann LeCun, and Alberto Bietti. The SSL interplay: Augmentations, inductive bias, and generalization. In *ICML*, 2023b.

Françoise Chatelin. *Spectral Approximation of Linear Operators*. Academic Press, 1983.

Ricky Chen and Yaron Lipman. Riemannian flow matching on general geometries, 2023. ArXiv preprint 2302.03660.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICLR*, 2020.

Fan Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

Ronald Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

Edward Condon and George Shortley. *The Theory of Atomic Spectra*. Cambridge University Press, 1935.

Heinz Otto Cordes. *Spectral Theory of Linear Differential Operators and Comparison Algebras*. Cambridge University Press, 1987.

Zhijie Deng, Jiaxin Shi, Hao Zhang, Peng Cui, Cewu Lu, and Jun Zhu. Neural eigenfunctions are structured representation learners, 2022. ArXiv preprint 2210.12637.

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped Langevin diffusion. In *ICLR*, 2022.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In *ICML*, 2021.

Fluid Dynamics. Georgii Ivanovich Petrov (on his 100th birthday). *Fluid Dynamics*, 47:289–291, 2012.

Christopher Frye and Costas Efthimiou. *Spherical Harmonics in p Dimensions*. World Scientific Publishing Company, 2014.

Aldo Glielmo, Brooke Husic, Alex Rodriguez, Cecilia Clementi, Frank Noé, and Alessandro Laio. Unsupervised learning methods for molecular simulation data. *Chemical Reviews*, 121(16):9722–9758, 2021.

Gene Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14:403–420, 1970.

Michael Greenacre. *Theory and Applications of Correspondence Analysis*. Academic Press, 1984.

Ulf Grenander and Michael Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 56(4):549–603, 1994.

Jihun Ham, Daniel Lee, Sebastian Mika, and Bernhard Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *ICML*, 2004.

Jiequn Han, Jianfeng Lu, and Mo Zhou. Solving high-dimensional eigenvalue problems using deep neural networks: A diffusion Monte Carlo like approach. *Journal of Computational Physics*, 423:109792, 2020.

Jeff HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. In *NeurIPS*, 2021.

Charles Harris, Jarrod Millman, Stéfan van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, 2020.

Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg. Graph Laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research*, 8:1325–1368, 2007.

Charles Hermite. Sur la formule d'interpolation de Lagrange. (Extrait d'une lettre de M. Hermite à M. Borchardt). *Journal für die reine und angewandte Mathematik*, 84:70–79, 1877.

Daniel Johnson, Ayoub El Hanchi, and Chris Maddison. Contrastive learning can find an optimal basis for approximately view-invariant functions. In *ICLR*, 2023.

Ian Jolliffe. *Principal Component Analysis*. Springer, 2002.

Vladimir Kondrachov. On certain properties of functions in the spaces L_p. *Doklady Akademii Nauk SSSR*, 48(8): 563–565, 1945.

Jason Lee, Qi Lei, Nikunj Saunshi, and Jiacheng Zhuo. Predicting what you already know helps: Provable self-supervised learning. In *NeurIPS*, 2021.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *NeurIPS*, 2016.

Giacomo Meanti, Luigi Carratino, Lorenzo Rosasco, and Alessandro Rudi. Kernel methods through the roof: Handling billions of points efficiently. In *NeurIPS*, 2020.

Stanislav Minsker. On some extensions of Bernstein's inequality for self-adjoint operators. *Statistics & Probability Letters*, 127:111–119, 2017.

Leon Mirsky. Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics*, 11(1):50–59, 1960.

Loucas Pillaud-Vivien and Francis Bach. Kernelized diffusion maps. In *COLT*, 2023.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NeurIPS*, 2007.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *NeurIPS*, 2017.

Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *NeurIPS*, 2015.

Bernhard Schölkopf and Alexander Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and beyond*. MIT Press, 2001.

Bernhard Schölkopf, Ralf Herbrich, and Alex Smola. A generalized representer theorem. In *COLT*, 2001.

Erich Schubert, Sibylle Hess, and Katharina Morik. The relationship of DBSCAN to matrix factorization and spectral clustering. In *Lernen, Wissen, Daten, Analysen*, 2018.

Mikhail Shubin. *Pseudodifferential Operators and Spectral Theory*. Springer, 1987.

Patrice Simard, Bernard Victorri, Yann LeCun, and John Denker. Tangent prop - a formalism for specifying selected invariances in an adaptive network. In *NeurIPS*, 1991.

Amit Singer. From graph to manifold laplacian: The convergence rate. *Applied and Computational Harmonic Analysis*, 21(1):128–134, 2006.

Josef Singer. On the equivalence of the Galerkin and Rayleigh-Ritz methods. *The Aeronautical Journal*, 66 (621):592, 1962.

Joel Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8 (1-2):1–230, 2015.

David van Dijk, Roshan Sharma, Juozas Nainys, Kristina Yim, Pooja Kathail, Ambrose Carr, Cassandra Burdziak, Kevin R. Moon, Christine Chaffer, Diwakar Pattabiraman, Brian Bierie, Linas Mazutis, Guy Wolf, Smita Krishnaswamy, and Dana Pe'er. Recovering gene interactions from single-cell data using data diffusion. *Cell*, 174(3): 716–729, 2018.

Charles van Loan. Generalizing the singular value decomposition. *Journal on Numerical Analysis*, 13(1):76–83, 1976.

Hermann Weyl. Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). *Journal on Numerical Analysis*, 74(4):441–479, 1912.

Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *NeurIPS*, 2000.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

Wei Zhang, Tiejun Li, and Christof Schütte. Solving eigenvalue PDEs of metastable diffusion processes using artificial neural networks. *Journal of Computational Physics*, 465:111377, 2022.

Ding-Xuan Zhou. Derivative reproducing properties for kernel methods in learning theory. *Journal of Computational and Applied Mathematics*, 220(1):456–463, 2008.

Hao Zhu and Piotr Koniusz. Generalized Laplacian eigenmaps. In *NeurIPS*, 2022.

Hao Zhu, Ke Sun, and Piotr Koniusz. Contrastive Laplacian eigenmaps. In *NeurIPS*, 2021.

Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *ICML*, 2003.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Subtleties. In this paper, we have explicitly dissociated between the way to estimate the quadratic form $\langle f, \mathcal{L}f\rangle$, and the space of functions to estimate its spectral decomposition. In particular, we have seen how the curse of dimensionality can be avoided when considering smooth spaces of functions that align well with eigenfunctions. We have also discussed the possibility to estimate the quadratic from associated with $\mathcal{L}$ through subsampling, or through finite differences. This creates a lot of different potential implementations under the name of graph Laplacians.

Limitations. It should be noted that, although it is likely, it has not been formally proven that the original graph Laplacian implementations suffer from minimax lower bound in $\Omega(n^{1/d})$ (or alike) over the class of operators with smooth eigenfunctions. Moreover, even if such lower bounds were proved, it does not rule out the highly unlikely possibility that there exists a single operator for which some graph Laplacian method convergences in $\Theta(n^{1/d})$ while our method does in $\Theta(n^{1/2})$ (or alike), while for all the other operators it beats our method. Finally, note that one can always define the class of functions for which a method with tuned hyper parameters "beats the curse of dimensionality". However, for the original graph Laplacians, these classes of functions are likely to be of low interest regarding operators encountered in real problems.

A Proofs

A.1 Characterization of $\mathcal{L}_0$

The characterization of $\mathcal{L}_0$ follows from multidimensional integration by part,

$$\begin{aligned}
\langle f, \mathcal{L}_0 g\rangle_{L^2(\rho)} &= \int f(x)(\mathcal{L}_0 g)(x)p(x)\,\mathrm{d}x = \mathbb{E}[\langle \nabla f(X), \nabla g(X)\rangle] \\
&= \lim_{r\to+\infty} \sum_{i\in[d]} \int_{\|x\|<r} \underbrace{\partial_i f(x)}_{u'} \underbrace{p(x)\partial_i g(x)}_{v}\,\mathrm{d}x \\
&= \lim_{r\to+\infty} \sum_{i\in[d]} \int_{\|x\|=r} \underbrace{f(x)}_{u} \underbrace{p(x)\partial_i g(x)}_{v} \langle e_i, n\rangle\,\mathrm{d}S - \int_{\|x\|<r} \underbrace{f(x)}_{u} \underbrace{\partial_i(p(x)\partial_i g(x))}_{v'}\,\mathrm{d}x \\
&= -\int f(x)\frac{1}{p(x)} \sum_{i\in[d]} \partial_i(p(x)\partial_i g(x))p(x)\,\mathrm{d}x,
\end{aligned}$$

where we used the fact, when p is regular enough, e.g. V is coercive, the surface integral goes to zero when g and f are smooth enough. As a consequence,

$$\begin{aligned}
(\mathcal{L}_0 f)(x) &= -\frac{1}{p(x)} \sum_{i\in[d]} \partial_i(p(x)\partial_i f(x)) = -\frac{1}{p(x)} \sum_{i\in[d]} p(x)\partial_{ii}^2 f(x) + \partial_i p(x)\partial_i f(x) \\
&= -\Delta f(x) - \frac{1}{p(x)} \langle \nabla p(x), \nabla f(x)\rangle = -\Delta f(x) - \langle \nabla \log p(x), \nabla f(x)\rangle \\
&= -\Delta f(x) + \langle \nabla V(x), \nabla f(x)\rangle .
\end{aligned}$$

A.2 Operator Details

Consider the mapping F, we have

$$\langle f, F\alpha\rangle_{L^2(\rho)} = \sum_{i\in[p]} \alpha_i \langle f, \varphi_i\rangle_{L^2(\rho)} = \left\langle (\langle f, \varphi_i\rangle_{L^2(\rho)})_{i\in[p]}, \alpha \right\rangle_{\mathbb{R}^p} = \langle F^* f, \alpha\rangle_{\mathbb{R}^p} .$$

As a consequence, with the adjunction with respect to $L^2(\rho)$ and $(\mathbb{R}^p, \ell^2)$

$$F^* : f \in L^2(\rho) \mapsto (\langle f, \varphi_i\rangle_{L^2(\rho)})_{i\in[p]} \in \mathbb{R}^p.$$

We deduce the characterization of Φ given in (6),

$$F^* F\alpha = (\langle F\alpha, \varphi_i\rangle_{L^2(\rho)})_{i\in[p]} = (\sum_{j\in[n]} \alpha_j \langle \varphi_j, \varphi_i\rangle_{L^2(\rho)})_{i\in[p]} = (\langle \varphi_j, \varphi_i\rangle_{L^2(\rho)})_{i,j\in[p]}\alpha = \Psi\alpha.$$

Similarly for the characterization of L in (5),

$$F^* \mathcal{L} G\alpha = (\langle \mathcal{L}G\alpha, \varphi_i\rangle_{L^2(\rho)})_{i\in[p]} = (\mathcal{E}_{\mathcal{L}}(G\alpha, \varphi_i))_{i\in[p]} = (\sum_{j\in[n]} \alpha_j \mathcal{E}_{\mathcal{L}}(\varphi_j, \psi_i))_{i\in[p]} = (\mathcal{E}_{\mathcal{L}}(\varphi_j, \psi_i))_{i,j\in[p]}\alpha = L\alpha.$$

A.3 Generalized Singular Value Decomposition

Let us consider three matrices L, Φ, Ψ in $\mathbb{R}^{p\times p}$. We would like to show that there exists three matrices A, B, S such that

$$ALB^\top = S, \qquad A\Phi A^\top = I, \quad B\Psi B^\top = I.$$

Remark that this generalized SVD is not the two-matrices version of van Loan (1976), but the weighted single-matrix one that appeared in correspondence analysis (Greenacre, 1984; Jolliffe, 2002).

To find such a decomposition, let us consider the singular value decomposition of

$$\Phi^{-1/2}L\Psi^{-1/2} = XSY^\top, \qquad X^\top X = Y^\top Y = I.$$

We have

$$X^\top\Phi^{-1/2}L\Psi^{-1/2}Y = S, \qquad (\Phi^{-1/2}X)^\top\Phi(\Phi^{-1/2}X) = (\Psi^{-1/2}Y)^\top\Psi(\Psi^{-1/2}Y) = I.$$

As a consequence, setting $A = (\Phi^{-1/2}X)^\top$ and $B = (\Psi^{-1/2}Y)$ we get

$$ALB^\top = S, \qquad A\Phi A^\top = B\Psi B^\top = I.$$

The first equation can equally be written, using that $A^{-1} = \Phi A^\top$,

$$ALB^\top = S \qquad \Leftrightarrow \qquad LB^\top = \Phi A^\top S \qquad \Leftrightarrow \qquad L = \Phi A^\top SB\Psi \qquad \Leftrightarrow \qquad AL = SB\Psi.$$

It can equally be expressed in term of columns as

$$[LB^\top]_i = L[B^\top]_i = L\beta_i = [\Phi A^\top S]_i = S_{ii}[\Phi A^\top]_i = S_{ii}\Phi[A^\top]_i = S_{ii}\Phi\alpha_i,$$

which matches the docstring formulation of Scipy for the symmetric case (with $\alpha_i = \beta_i = \mathtt{v}$, $S_{ii} = \mathtt{w}$, $L = \mathtt{a}$, $\Phi = \mathtt{b}$):

`https://github.com/scipy/scipy/blob/v1.11.3/scipy/linalg/_decomp.py#L325`.

The advantage of using the GSVD rather than the SVD of the system $\Psi^{-1/2}L\Phi^{-1/2}$ is that it requires less flops (although the big-O complexity will be the same). The GSVD is roughly equivalent to one matrix inversion instead of three if we choose to first invert Ψ and Φ before performing one SVD (Golub and Reinsch, 1970).

A.4 Proof of Theorem 1

Lemma 2 (Error decomposition)**.** *With the $(\hat\lambda_i, \hat f_i, \hat g_i)$ obtained with Algorithm 1, when $L^{-1}(\hat L - L) - I/2$ is positive, for any $a \in [0,1]$,*

$$\left\|\mathcal{L}^{-1} - (\sum_{i\in[p]}\hat\lambda_i\hat f_i\otimes\hat g_i)^{-1}\right\| \le \left\|(I-\Pi_G)\mathcal{L}^{-1}\right\| + \left\|\mathcal{L}^{-1}(I-\Pi_F)\right\| + \gamma\left\|\Psi^{1/2}L^{-a}\right\|\left\|L^{-a}\Phi^{1/2}\right\|\left\|L^{-(1-a)}(\hat L - L)L^{-(1-a)}\right\|,$$

where $\gamma = 1 + 2(\|L\|\left\|L^{-1}\right\|)^{\min(a,1-a)}$.

Proof. We start with

$$\left\|\mathcal{L}^{-1} - (\sum_{i\in\mathbb{N}}\hat\lambda_i\hat f_i\otimes\hat g_i)^{-1}\right\| \le \left\|\mathcal{L}^{-1} - (\Pi_F\mathcal{L}\Pi_G)^{-1}\right\| + \left\|(\Pi_F\mathcal{L}\Pi_G)^{-1} - \sum_{i\in\mathbb{N}}\hat\lambda_i^{-1}\hat g_i\otimes\hat f_i\right\|.$$

The first term is rewritten with

$$\begin{aligned}\left\|\mathcal{L}^{-1} - (\Pi_F\mathcal{L}\Pi_G)^{-1}\right\| &= \left\|\mathcal{L}^{-1} - \Pi_G\mathcal{L}^{-1}\Pi_F\right\| = \left\|(I-\Pi_G)\mathcal{L}^{-1} + \Pi_G\mathcal{L}^{-1}(I-\Pi_F)\right\| \\ &\le \left\|(I-\Pi_G)\mathcal{L}^{-1}\right\| + \left\|\Pi_G\mathcal{L}^{-1}(I-\Pi_F)\right\| \\ &\le \left\|(I-\Pi_G)\mathcal{L}^{-1}\right\| + \left\|\mathcal{L}^{-1}(I-\Pi_F)\right\|.\end{aligned}$$

The second term is rewritten with

$$\begin{aligned}\left\|(\Pi_F\mathcal{L}\Pi_G)^{-1} - \sum_{i\in\mathbb{N}}\hat\lambda_i^{-1}\hat g_i\otimes\hat f_i\right\| &= \left\|G(F^*\mathcal{L}G)^{-1}F^* - \sum_{i\in\mathbb{N}}\hat\lambda_i^{-1}\hat g_i\otimes\hat f_i\right\| = \left\|G(L^{-1} - \hat L^{-1})F^*\right\| \\ &= \left\|\Psi^{1/2}(L^{-1} - \hat L^{-1})\Phi^{1/2}\right\|.\end{aligned}$$

The difference between the inverse can be worked out as,

$$\begin{aligned} L^{-1} - \hat{L}^{-1} &= L^{-1}(\hat{L} - L)\hat{L}^{-1} = L^{-1}(\hat{L} - L)(L^{-1} + \hat{L}^{-1} - L^{-1}) \\ &= L^{-1}(\hat{L} - L)L^{-1} - L^{-1}(\hat{L} - L)(L^{-1} - \hat{L}^{-1}) \\ &= (I + L^{-1}(\hat{L} - L))^{-1}L^{-1}(\hat{L} - L)L^{-1}, \end{aligned}$$

where the last equality is true when the matrix $I + L^{-1}(\hat{L} - L)$ is invertible, which is notably implied by $L^{-1}(\hat{L} - L) \succeq -I$. As a consequence,

$$\begin{aligned} \Psi^{1/2}(L^{-1} - \hat{L}^{-1})\Phi^{1/2} &= \Psi^{1/2}L^{-1}(\hat{L} - L)L^{-1}\Phi^{1/2} + \Psi^{1/2}L^{-1}(\hat{L} - L)(\hat{L}^{-1} - L^{-1})\Phi^{1/2} \\ &= \Psi^{1/2}L^{-1}(\hat{L} - L)L^{-1}\Phi^{1/2} + \Psi^{1/2}L^{-1}(\hat{L} - L)(I + L^{-1}(\hat{L} - L))^{-1}L^{-1}(\hat{L} - L)L^{-1}\Phi^{1/2}. \end{aligned}$$

We can translate this last equality in operator norm: for any $a, b \in [0, 1]$

$$\begin{aligned} \left\|\Psi^{1/2}(L^{-1} - \hat{L}^{-1})\Phi^{1/2}\right\| \leq \left\|\Psi^{1/2}L^{-a}\right\| \left\|L^{-b}\Phi^{1/2}\right\| \\ \times \left(\left\|L^{-(1-a)}(\hat{L} - L)L^{-(1-b)}\right\| + \left\|L^{-(1-a)}(\hat{L} - L)(I + L^{-1}(\hat{L} - L))^{-1}L^{-1}(\hat{L} - L)L^{-(1-b)}\right\|\right). \end{aligned}$$

The last term can be bounded either with

$$\begin{aligned} &\left\|L^{-(1-a)}(\hat{L} - L)(I + L^{-1}(\hat{L} - L))^{-1}L^{-1}(\hat{L} - L)L^{-(1-b)}\right\| \\ &= \left\|L^{a}L^{-1}(\hat{L} - L)(I + L^{-1}(\hat{L} - L))^{-1}L^{-a}L^{-(-1-a)}(\hat{L} - L)L^{-(1-b)}\right\| \\ &\leq \|L^a\| \left\|L^{-a}\right\| \left\|L^{-1}(\hat{L} - L)(I + L^{-1}(\hat{L} - L))^{-1}\right\| \left\|L^{-(1-a)}(\hat{L} - L)L^{-(1-b)}\right\|, \end{aligned}$$

or with,

$$\begin{aligned} &\left\|L^{-(1-a)}(\hat{L} - L)(I + L^{-1}(\hat{L} - L))^{-1}L^{-1}(\hat{L} - L)L^{-(1-b)}\right\| \\ &= \left\|L^{-(1-a)}(\hat{L} - L)L^{-(1-b)}L^{1-b}(I + L^{-1}(\hat{L} - L))^{-1}L^{-1}(\hat{L} - L)L^{-(1-b)}\right\| \\ &\leq \left\|L^{(1-b)}\right\| \left\|L^{-(1-b)}\right\| \left\|L^{-1}(\hat{L} - L)(I + L^{-1}(\hat{L} - L))^{-1}\right\| \left\|L^{-(1-a)}(\hat{L} - L)L^{-(1-b)}\right\|. \end{aligned}$$

Using the fact that $\left\|(I + A)^{-1}A\right\| \leq 2$ as soon as $A + I/2 \succeq 0$, we deduce that as soon as $L^{-1}(\hat{L} - L) \succeq -I/2$,

$$\left\|\Psi^{1/2}(L^{-1} - \hat{L}^{-1})\Phi^{1/2}\right\| \leq \left\|\Psi^{1/2}L^{-a}\right\| \left\|L^{-b}\Phi^{1/2}\right\| \left\|L^{-(1-a)}(\hat{L} - L)L^{-(1-b)}\right\| \left(1 + 2(\|L\| \left\|L^{-1}\right\|)^{\min(a,1-b)}\right).$$

This explains the decomposition in the lemma. □

We continue by bounding the empirical versus population difference $\left\|L - \hat{L}\right\|$. To do so, we will use Bernstein concentration inequality.

Lemma 3. *Let $A(x)$ be a $p \times p$ matrix bounded by M and $B_1 = \mathbb{E}[A(X)^\top A(X)]$, $B_2 = \mathbb{E}[A(X)^\top A(X)]$. For $t \geq 0$,*

$$\mathbb{P}_{(x_k)}\left(\left\|\mathbb{E}[A(X)] - \frac{1}{n}\sum_{k\in[n]} A(x_k)\right\| \geq t\right) \leq 4\frac{\operatorname{Tr} B_1 + B_2}{\max \|B_i\|} \exp\left(\frac{-3nt^2}{6\max(\|B_i\|) + 2Mt}\right).$$

For any $t \in [0, M]$,

$$\mathbb{P}_{(x_k)}\left(\left\|\mathbb{E}[A(X)] - \frac{1}{n}\sum_{k\in[n]} A(x_k)\right\| \geq t\right) \leq 2p \exp\left(\frac{-3nt^2}{8M^2}\right).$$

Proof. This is Theorem 7.3.1 and Theorem 6.1.1 of Tropp (2015). □

Lemma 4 (Estimation error)**.** *With n data points, our algorithm guarantees, for $t < p^2 H_\infty^2$,*

$$\mathbb{P}(\left\|L - \hat{L}\right\| > t) \leq 2p \exp(-\frac{3nt^2}{8p^2 H_\infty^2}),$$

where $H_\infty = \sup_{i,j,x} H(\varphi_i, \psi_j, x)$. In other terms, for any $\delta \in (0,1)$ and $n > 3\log(2p/\delta)$, with probability $1-\delta$

$$\|L - \hat{L}\| \leq \sqrt{\frac{8p^2 H_\infty^2}{3n} \log(2p/\delta)}.$$

Proof. We would like to use Bernstein inequality with L and $\hat{L}$, they are both built from the matrix

$$A(x)_{ij} = H(\varphi_i, \psi_j, x).$$

Without specific structure on H, we proceed with the following bound

$$\|A(x)\|_{\mathrm{op}}^2 \leq \|A(x)\|_2^2 \leq \sum_{i,j} H(\varphi_i, \psi_j, x)^2 \leq p^2 \sup H(\varphi_i, \psi_j, x)^2 =: p^2 H_\infty^2.$$

As a consequence, we have the following bound

$$\mathbb{P}(\left\|L - \hat{L}\right\| > t) \leq 2p \exp(-\frac{3nt^2}{8M^2}), \qquad \text{with} \qquad M = p^2 H_\infty^2.$$

To parse the bound more easily, let us invert t, we set

$$\delta = 2p \exp(-\frac{3nt^2}{8M^2}), \qquad t = \sqrt{\frac{8M^2}{3n} \log(2p/\delta)}.$$

When n is big enough to ensure $t < M$, i.e., $n > 3\log(2p/\delta)$, we have that with probability $1-\delta$

$$\left\|L - \hat{L}\right\| \leq \sqrt{\frac{8M^2}{3n} \log(2p/\delta)}.$$

Replacing M by pH_∞ ends the proof. □

The proof of the theorem follows directly from the previous lemmas.

A.5 Proof of Theorem 2

In order to prove Theorem 2, we need to specify the values taken by $\|\mathcal{L}(I - \Pi_F)\|$ and $\|(I - \Pi_G)\mathcal{L}\|$. In all the following, it is useful to introduce

$$\hat{\Sigma} = p^{-1} \sum_{i \in [p]} \varphi_i \varphi_i^*, \qquad \text{and} \qquad \hat{\Xi} = p^{-1} \sum_{i \in [p]} \psi_i \psi_i^*. \tag{22}$$

Lemma 5. *Let $\mathcal{A}$ be some operator in $L^2(\rho)$. Assume that the $(\varphi_i)_{i \in [p]}$ were chosen independently at random according to the same distribution $\mu \in \Delta_{L^2(\rho)}$, with $\Sigma = \mathbb{E}_{\varphi \sim \mu}[\varphi \varphi^*]$. Assume that there exists $a \in [0,1]$ and $M > 0$ such that $\left\|\Sigma^{-a/2}\varphi\right\|_{L^2(\rho)} \leq M^{1/2}$ independently of φ. Let Π_* denotes a κ-dimensional projection on an eigenspace of Σ and Σ_* the associated restriction $\Sigma_* = \Pi_* \Sigma$. For any $\delta \in (0,1)$ for any $p > 3\log(2\kappa/\delta)$, it holds with probability $1-\delta$,*

$$\|(I - \Pi_F)\mathcal{A}\| \leq \|(I - \Pi_*)\mathcal{A}\| + \left\|\Sigma_*^{-(1-a)}\mathcal{A}\right\| \sqrt{\frac{8M^2}{3p} \log(2\kappa/\delta)}.$$

Here, A^{-1} denotes the pseudo-inverse of A.

Proof. For simplicity, we denote $\Pi = \Pi_F$ in the proof. We split the error with

$$\|(I - \Pi)\mathcal{A}\| = \|(I - \Pi)(I - \Pi_*)\mathcal{A} + (I - \Pi)\Pi_*\mathcal{A}\| \leq \|(I - \Pi_*)\mathcal{A}\| + \|(I - \Pi)\Pi_*\mathcal{A}\|.$$

The first term depends on assumptions on the problem, while the second term depends on the empirical approximation of Π_* by Π. Using the fact that, with $A^\dagger$ denoting the pseudo-inverse,

$$A^\dagger - B^\dagger = A^\dagger(B-A)B^\dagger - (I-\Pi_A)B^\dagger + A^\dagger(I-\Pi_B),$$

we have

$$\begin{aligned}\Pi_* - \Pi &= \Sigma_*^\dagger\Sigma - \hat\Sigma^\dagger\hat\Sigma = \Sigma_*^\dagger(\Sigma-\hat\Sigma) + (\Sigma_*^\dagger - \hat\Sigma^\dagger)\hat\Sigma \\ &= \Sigma_*^\dagger(\Sigma-\hat\Sigma) - \Sigma_*^\dagger(\Sigma-\hat\Sigma)\hat\Sigma^\dagger\hat\Sigma - (I-\Pi_*)\hat\Sigma^\dagger\hat\Sigma + \Sigma_*^\dagger(I-\Pi)\hat\Sigma. \\ &= \Sigma_*^\dagger(\Sigma-\hat\Sigma)(I-\Pi) - (I-\Pi_*)\Pi = \Sigma_*^\dagger(\Sigma-\hat\Sigma)(I-\Pi) + (\Pi_*-\Pi)\Pi.\end{aligned}$$

As a consequence

$$\mathcal{A}^*\Pi_*(I-\Pi) = \mathcal{A}^*(\Pi_*-\Pi)(I-\Pi) = \mathcal{A}^*\Sigma_*^\dagger(\Sigma-\hat\Sigma)(I-\Pi). = \mathcal{A}^*\Sigma_*^\dagger\Pi_*(\Sigma-\hat\Sigma)(I-\Pi),$$

which we translate in operator norm with

$$\|(I-\Pi)\Pi_*\mathcal{A}\| \leq \left\|\Sigma_*^{-(1-a)}\mathcal{A}\right\| \left\|\Pi_*\Sigma^{-a}(\Sigma-\hat\Sigma)\right\|.$$

We now need to bound the difference between Σ and $\hat\Sigma$. This is a simple application of Bernstein with $\Pi_*\Sigma^{-a}\varphi(X)\varphi(X)^\top$. In particular, if $\left\|\Sigma^{-a/2}\varphi\right\|_{L^2(\rho)} \leq M^{1/2}$, we have for $t \in [0, M]$,

$$\mathbb{P}(\left\|\Pi_*\Sigma^{-a}(\Sigma-\hat\Sigma)\right\| > t) \leq 2\kappa\exp(-\frac{3pt^2}{8M^2}).$$

Once again, let us invert t, we set

$$\delta = 2\kappa\exp(-\frac{3pt^2}{8M^2}), \qquad t = \sqrt{\frac{8M^2}{3p}\log(2\kappa/\delta)}.$$

When p is big enough to ensure $t < M$, in particular, when $p > 3\log(2\kappa/\delta)$, we have that with probability $1-\delta$

$$\left\|\Pi_*\Sigma^{-a}(\Sigma-\hat\Sigma)\right\| \leq \sqrt{\frac{8M^2}{3p}\log(2\kappa/\delta)}.$$

This ends the proof of the lemma. □

The proof of the theorem follows directly from the previous lemma. In the setting of Theorem 2, it is interesting to specify the value of $\left\|L^{-a}\Psi^{1/2}\right\|$.

Lemma 6. *Let $\|\psi\|_{L^2(\rho)} \leq M^{1/2}$ and $\left\|\Xi^{-1/2}\psi\right\|_{L^2(\rho)} \leq M_\infty$ with $M_\infty > 1/2$. For any $a \in [0,1]$ if L is symmetric, or for $a = 1$ and any L, for any $\delta \in (0,1)$ and $\kappa \in \mathbb{N}$, when*

$$p \geq \max(11M_\infty^2\log(2\kappa/\delta), 5M^2\log(\frac{2M\mu_{\kappa+1}}{\|\Xi\|})),$$

it holds with probability at least $1-\delta$,

$$\left\|L^{-a}\Psi^{1/2}\right\| \leq 2^a M^{(1-a)/2}\left(\left\|\Xi^{-1/2}\mathcal{L}^{-1}\right\| + \mu_{\kappa+1}^{-1/2}\right)^a,$$

where μ_κ is the κ-th eigenvalue of Ξ.

Proof. Note that there exists two isometric mappings $U : L^2(\rho) \to \mathbb{R}^p$ and $V : L^2(\rho) \to \mathbb{R}^p$ such that

$$F = U\hat\Sigma^{1/2}, \qquad \text{and} \qquad G = V\hat\Xi^{1/2},$$

and $UU^* = VV^* = I$. As a consequence

$$L^{-a}\Psi^{1/2} = (F^*\mathcal{L}G)^{-a}(F^*F)^{1/2} = V\hat\Xi^{-a/2}\mathcal{L}^{-a}\hat\Sigma^{-a/2}\hat\Sigma^{1/2}U^*,$$

which we translate in operator norm with

$$\left\|L^{-a}\Psi^{1/2}\right\| = \left\|\hat{\Xi}^{-a/2}\mathcal{L}^{-a}\hat{\Sigma}^{(1-a)/2}\right\| \leq \left\|\hat{\Xi}^{-1/2}\mathcal{L}^{-1}\right\|^a \left\|\hat{\Sigma}\right\|^{(1-a)/2},$$

where we have used the fact that $\|A^aB^a\| \leq \|AB\|^a$ is A and B are positive (Cordes, 1987). For the last term, we know that

$$\left\|\hat{\Sigma}\right\| \leq \operatorname{Tr}(\hat{\Sigma}) = n^{-1}\sum_{i\in[n]}\|\varphi_i\|^2 \leq M.$$

For the first term, considering Π_* a projection on the top κ eigen functions of Σ, we have

$$\left\|\hat{\Xi}^{-1/2}\mathcal{L}^{-1}\right\| \leq \left\|\hat{\Xi}^{-1/2}\Pi_*\mathcal{L}^{-1}\right\| + \left\|\hat{\Xi}^{-1/2}(I-\Pi_*)\mathcal{L}^{-1}\right\| \qquad \leq \left\|\hat{\Xi}^{-1/2}\Pi_*\mathcal{L}^{-1}\right\| + \left\|\hat{\Xi}^{-1/2}(I-\Pi_*)\right\|\left\|\mathcal{L}^{-1}\right\|.$$

The first part will concentrate with

$$\begin{aligned}\left\|\hat{\Xi}^{-1/2}\Pi_*\mathcal{L}^{-1}\right\|^2 = \left\|\mathcal{L}^{-1}\Pi_*\hat{\Xi}^{-1}\Pi_*\mathcal{L}^{-1}\right\| &= \left\|\mathcal{L}^{-1}\Xi^{-1/2}\Xi^{1/2}\Pi_*\hat{\Xi}^{-1}\Pi_*\Xi^{1/2}\Xi^{-1/2}\mathcal{L}^{-1}\right\| \\ &\leq \left\|\mathcal{L}^{-1}\Xi^{-1/2}\right\|^2\left\|\Pi_*\Xi^{1/2}\hat{\Xi}^{-1}\Xi^{1/2}\Pi_*\right\|.\end{aligned}$$

We have already seen how to treat the last term

$$\begin{aligned}\Xi^{1/2}\hat{\Xi}^{-1}\Xi^{1/2} &= I + \Xi^{1/2}(\hat{\Xi}^{-1}-\Xi^{-1})\Xi^{1/2} = I + \Xi^{-1/2}(\Xi-\hat{\Xi})\Xi^{-1/2}\Xi^{1/2}\hat{\Xi}^{-1}\Xi^{1/2} \\ &= (I-\Xi^{-1/2}(\Xi-\hat{\Xi})\Xi^{-1/2})^{-1}.\end{aligned}$$

Using Bernstein inequality, we deduce that, if $M_\infty > 1/2$,

$$\mathbb{P}(\left\|\Pi_*\Xi^{-1/2}(\Xi-\hat{\Xi})\Xi^{-1/2}\right\| > 1/2) \leq 2\kappa\exp(-\frac{3p}{32M_\infty^2}).$$

Let us invert the relation, we want

$$\delta \leq 2\kappa\exp(-\frac{3p}{32M_\infty^2}), \qquad p \geq \frac{32M_\infty^2}{3}\log(2\kappa/\delta).$$

For the second part, we proceed with

$$\begin{aligned}\left\|\hat{\Xi}^{-1/2}(I-\Pi_*)\right\|^2 = \left\|(I-\Pi_*)\hat{\Xi}^{-1}(I-\Pi_*)\right\| &\leq \left\|(I-\Pi_*)\Xi^{-1}(I-\Pi_*)\right\| + \left\|(I-\Pi_*)(\hat{\Xi}^{-1}-\Xi^{-1}(I-\Pi_*)\right\| \\ &\leq \left\|(I-\Pi_*)\Xi^{-1/2}\right\|^2 + \left\|(I-\Pi_*)\Xi^{-1}\right\|\left\|\Xi-\hat{\Xi}\right\|\left\|\Xi^{-1}(I-\Pi_*)\right\|,\end{aligned}$$

which can be rewritten as

$$\left\|\hat{\Xi}^{-1/2}(I-\Pi_*)\right\|^2 \leq \left\|(I-\Pi_*)\Xi^{-1/2}\right\|^2\left(1+\left\|\Xi-\hat{\Xi}\right\|\left\|\Xi^{-1/2}(I-\Pi_*)\right\|^2\right) \leq \frac{\mu_{\kappa+1}^{-1}}{1-\mu_{\kappa+1}^{-1}\left\|\Xi-\hat{\Xi}\right\|}.$$

Using Bernstein inequality in Hilbert space (Minsker, 2017), we have

$$\mathbb{P}(\left\|\Xi-\hat{\Xi}\right\| > t) \leq 2\frac{\operatorname{Tr}\Xi}{\|\Xi\|}\exp(-\frac{3pt^2}{8M^2}) \leq \frac{2M}{\|\Xi\|}\exp(-\frac{3pt^2}{8M^2}).$$

In particular, when

$$\frac{2M}{\|\Xi\|}\exp(-\frac{27p}{128M^2}) \leq \mu_{\kappa+1}^{-1} \qquad \Leftrightarrow \qquad \frac{128M^2}{27}\log(\frac{2M\mu_{\kappa+1}}{\|\Xi\|}) \leq p$$

we have that $\left\|\hat{\Xi}^{-1/2}(I-\Pi_*)\right\| \leq 2\mu_{\kappa+1}^{-1/2}$. Collecting the different pieces proves the lemma. □

A.6 Proof of Theorem 3

To prove Theorem 3, we start by reworking the estimation error between $\hat{L}$ and L.

Lemma 7 (Estimation error for $\mathcal{L}_0$)**.** *When $\mathcal{L} = \mathcal{L}_0$ and $\varphi_i = \psi_i = k_{x_i}$ with $k_x(y) = ((1 + x^\top y)/(1 + M))^s$ for M an almost sure upper bound on $\|X\|$, with n data points, our algorithm guarantees, for $t < s^2$,*

$$\mathbb{P}(\left\|L - \hat{L}\right\| > t) \leq 2\kappa \exp(-\frac{3nt^2}{8s^2}), \qquad \textit{with} \qquad \kappa = \binom{d+s}{s}.$$

As a consequence, for any $\delta \in (0,1)$ and $n > 3\log(2\kappa/\delta)$, it holds with probability $1-\delta$

$$\frac{\left\|L^{-1}\right\|}{1 - \|L^{-1}\|\,\|L - \hat{L}\|}\|\Phi\|^{-1/2}\|\Psi\|^{-1/2}\|L^{-1}\|\|L - \hat{L}\| \leq 2\left\|L^{-1}\right\|^2\sqrt{\frac{8s^2}{3n}\log(2\kappa/\delta)}.$$

Proof. The proof follows the one of Lemma 3 with

$$A(x)_{ij} = H(\varphi_i, \psi_j, x).$$

Without specific structure on H, we proceeded with $\|A(x)\| \leq pH_\infty$. In the specific case of $\mathcal{L}_0$, with $\mathcal{F} = \mathcal{G} = \mathcal{H}$, we get

$$\begin{aligned}\|A(x)\| &= \sup_{\|c\|=1} c^\top A(x)c = \sup_{\|c\|=1}\sum_{ij} c_i \left\langle \nabla k_{x_i}(x), \nabla k_{x_j}(x)\right\rangle c_j \\ &= \sup_{\|c\|=1} \|\sum_i c_i \nabla k_{x_i}(x)\|^2 \leq \sup_{\|c\|=1}\sum_i c_i^2 \left\|\nabla k_{x_i}(x)\right\|^2 \leq \sup\left\|\nabla k_x(y)\right\|^2.\end{aligned}$$

When $\mathcal{H}$ is defined from the kernel

$$k_x(y) = \left(\frac{1 + x^\top y}{1 + M}\right)^s \leq 1,$$

this becomes

$$\|A(x)\| \leq \sup\|\nabla k_x(y)\|^2 = \sup s^2\left(\frac{1 + x^\top y}{1 + M}\right)^{2(s-1)}\|x/(1 + M)\|^2 \leq s^2.$$

Similarly, this choice of k guarantees

$$\left\|\Phi^{1/2}\right\|^2 = \|\Phi\| \leq \sup\|k_x(y)\| \leq 1.$$

As a consequence, we have the following bound

$$\mathbb{P}(\left\|L - \hat{L}\right\| > t) \leq 2\kappa\exp(-\frac{3nt^2}{8s^4}),$$

where we have replaced p by κ since we $\mathrm{rank}(F) \leq \kappa$. The end of the proof is similar to the proof of Lemma 3. □

We continue by reworking the approximation error.

Lemma 8. *When $\mathcal{L}_p = \sum_{i\in[p]}\lambda_i f_i g_i^*$ is diagonalized by all the $\kappa = \binom{d+s}{d}$ polynomials of degree less or equal than s, $\|x\| < M$ almost surely and $k_x(y) = ((1 + x^\top y)/(1 + M))^s$, then if the φ_i are almost surely linearly independent,*

$$\left\|(I - \Pi_G)\mathcal{L}^{-1}\right\| \leq \lambda_{\kappa+1}^{-1}, \qquad \textit{and} \qquad \left\|\Pi_G\mathcal{L}^{-1}(I - \Pi_F)\right\| = 0.$$

Proof. In this case, the φ_i's span all the polynomials of degree less or equal than s, hence $(I - \Pi_F)\mathcal{L}_p = 0$. □

We continue by working out the terms $\left\|\Phi^{1/2}L^{-1}\right\|$, $\left\|L^{-1}\Psi\right\|$ and $\left\|L^{-1}\right\|$ appearing in Theorem 1.

Lemma 9. *When $\varphi_i = \psi_i$ and $\left\|\Sigma^{-1/2}\varphi\right\|_{L^2(\rho)} \leq M_\infty$ with $M_\infty > 1/2$, and $\kappa = \mathrm{rank}(\Xi)$. For any $\delta \in (0,1)$*

$$p \geq 11M_\infty^2\log(2\kappa/\delta),$$

it holds with probability at least $1-\delta$,

$$\max(\left\|\Psi^{1/2}L^{-1}\right\|, \left\|L^{-1}\Phi^{1/2}\right\|) \leq 2\left\|\Sigma^{-1/2}\mathcal{L}^{-1}\right\|, \qquad \textit{and} \qquad \left\|L^{-1}\right\| \leq 4\left\|\Sigma^{-1/2}\mathcal{L}^{-1}\Sigma^{-1/2}\right\|.$$

Proof. As detailed in the proof of Lemma 6,

$$\left\|L^{-1}\Psi^{1/2}\right\|^2 = \left\|\hat{\Xi}^{-1/2}\mathcal{L}^{-1}\right\|^2 = \left\|\mathcal{L}^{-1}\hat{\Xi}^{-1}\mathcal{L}^{-1}\right\| = \left\|\mathcal{L}^{-1}\Xi^{-1/2}\Xi^{1/2}\hat{\Xi}^{-1}\Xi^{1/2}\Xi^{-1/2}\mathcal{L}^{-1}\right\|$$
$$\leq \left\|\mathcal{L}^{-1}\Xi^{-1/2}\right\|^2 \left\|\Xi^{1/2}\hat{\Xi}^{-1}\Xi^{1/2}\right\| = \left\|\mathcal{L}^{-1}\Xi^{-1/2}\right\|^2 \left\|(I - \Xi^{-1/2}(\Xi - \hat{\Xi})\Xi^{-1/2})^{-1}\right\|.$$

Using Bernstein inequality, we deduce that, if $M_\infty > 1/2$,

$$\mathbb{P}(\left\|\Xi^{-1/2}(\Xi - \hat{\Xi})\Xi^{-1/2}\right\| > 1/2) \leq 2\kappa \exp(-\frac{3p}{32M_\infty^2}).$$

We invert the relation with

$$\delta \leq 2\kappa \exp(-\frac{3p}{32M_\infty^2}), \qquad p \geq \frac{32M_\infty^2}{3}\log(2\kappa/\delta).$$

Similarly, under the same event

$$\left\|L^{-1}\right\| = \left\|\hat{\Sigma}^{-1/2}\mathcal{L}^{-1}\hat{\Sigma}^{-1/2}\right\| \leq \left\|\hat{\Sigma}^{-1/2}\Sigma^{1/2}\right\|^2 \left\|\Sigma^{-1/2}\mathcal{L}^{-1}\Sigma^{-1/2}\right\| \leq 4\left\|\Sigma^{-1/2}\mathcal{L}^{-1}\Sigma^{-1/2}\right\|.$$

This explains the results of the lemma. □

Proof of Theorem 3. Combining the previous lemmas, we have refinement of Theorems 1 and 2 that when

$$n \geq 6\max(s^2\left\|\mathcal{L}^{-1}\right\|^{-2}, 1)\log(2\kappa/\delta), \qquad p \geq 11 \operatorname*{ess\,sup}_{\varphi} \left\langle \varphi, \Sigma^{-1}\varphi\right\rangle_{L^2(\rho)} \log(2\kappa/\delta),$$

it holds with probability $1 - 2\delta$,

$$\left\|\mathcal{L}^{-1} - G\hat{L}^{-1}F^*\right\| \leq \lambda_{\kappa+1}^{-1} + 12\|\Sigma^{-1/2}\mathcal{L}^{-1}\|^2\sqrt{\frac{8s^2}{3n}\log(\frac{2p}{\delta})}.$$

To end the proof of the theorem, notice that when $\mathcal{L}^{-1}$ is compact, the λ_i^{-1} are summable hence they are bounded by $ci^{-\tau}$ for some constant c and $\tau \geq 1$. Moreover, choosing

$$\kappa = \binom{s+d}{s} = \binom{d+s}{d} = \prod_{i\in[s]} \frac{(d+s-i+1)}{s-i+1)},$$

because for all $k \in [s]$,

$$\frac{d+s}{s} \leq \frac{d+k}{k} \leq d,$$

leads to

$$(s/d)^d \leq (1+s/d)^d \leq \max((1+d/s)^s, (1+s/d)^d) \leq p \leq \min(d^s, s^d) \leq s^d.$$

Hence our bound becomes

$$\left\|\mathcal{L}^{-1} - G\hat{L}^{-1}F^*\right\| \leq \lambda_{\kappa+1}^{-1} + 12\|\Sigma^{-1/2}\mathcal{L}^{-1}\|^2\sqrt{\frac{8s^2}{3n}\log(\frac{2p}{\delta})} \leq cd^{\tau d}s^{-\tau d} + \frac{20s\left\|\Sigma^{-1/2}L^{-1}\right\|^2}{n^{1/2}}\sqrt{\log\left(\frac{p}{\delta}\right)}.$$

Choosing $s = n^{1/2(\tau d+1)}$ leads to a bound $O(n^{-\tau d/2(\tau d+1)}\log(n))$, which holds as long as

$$n \geq 6\max(n^{1/(\tau d+1)}\left\|\mathcal{L}^{-2}\right\|, 1)\left(\frac{d}{2(\tau d+1)}\log(n) + \log(2/\delta)\right),$$

and using that $\left\|K^{-1}\varphi\right\| \leq \mu_{\kappa+1}^{-1}\|\varphi\|$,

$$n \geq p \geq 11cn^{\tau d/2(\tau d+1)}M\left(\frac{d}{2(\tau d+1)}\log(n) + \log(2/\delta)\right).$$

Both conditions can hold true for $p = n^{1/2}$ and $n > N$ for some $N \in \mathbb{N}$. The theorem in the main text considers the worst case where $\tau = 1$. □

	Graph-Laplacian	Kernel-Laplacian
Time complexity	$n^2d + p^2n$	$pnd + p^2n$
Memory complexity	n^2	pn

Table 1: Computational complexity of graph-Laplacian and kernel-Laplacian. Not only kernel-Laplacian does not suffer from the curse of dimension, which contrasts with graph-Laplacian, but it does so without requiring extra computations (recall that $p \leq n$ is taken as a small integer).

A.7 Variance of a gradient estimate through Jacobian vector products

Consider the estimator

$$\|\nabla f(X)\|^2 = \mathbb{E}_U[c\ \langle \nabla f(X), U\rangle^2],$$

for U uniform on the sphere. The proportionality constant is given by

$$c = \mathbb{E}_U[\langle e_1, U\rangle^2]^{-1} = \mathbb{E}_U[U_1^2]^{-1} = \int_0^1 u_1^2\sqrt{1-u_1^2}\,\mathrm{Vol}(\mathbb{S}^{d-2})\,\mathrm{d}u_1 = \frac{\pi}{16}\,\mathrm{Vol}(\mathbb{S}^{d-2}),$$

which grows exponentially fast with respect to the input dimension d. The estimator $G = c\ \langle \nabla f(X), U\rangle^2$ has a second moment, which assuming without restriction that $\|\nabla f(X)\|^2 = 1$ is equal to

$$\begin{aligned}\mathbb{E}[G^2] &= c^2\mathbb{E}_U[\langle e_1, U\rangle^4]^{-1} \geq \frac{c^2}{16}\mathbb{P}(|U_1| > 1/2) = \frac{c^2}{8\,\mathrm{Vol}(\mathbb{S}^{d-1})}\int_{1/2}^1 \sqrt{1-u_1^2}\,\mathrm{Vol}(\mathbb{S}^{d-2})\,\mathrm{d}u_1 \\ &= \frac{c^2\,\mathrm{Vol}(\mathbb{S}^{d-2})}{8\,\mathrm{Vol}(\mathbb{S}^{d-1})}\left(\frac{\pi}{6} - \frac{\sqrt{3}}{8}\right),\end{aligned}$$

while its mean is equal to one, hence its variance will grow exponentially fast as the dimension d increases. This explains the usefulness to restrict the loss (19) to a few tangent directions, allowing to lower the variance of stochastic gradient descent and to accelerate its convergence (Bubeck, 2015).

B Additional Experiments and Details

B.1 Graph-Laplacian Implementation

Many studies of graph-Laplacian are set in transductive settings, restricting $\mathcal{X}$ to a finite number of points (Belkin and Niyogi, 2003; Zhu et al., 2003). In this study, we consider graph-Laplacian as a proxy to estimate $\mathcal{E}_\mathcal{L}$ empirically as per (16). As such, we can use Galerkin method with graph-Laplacian, only solving a eigenvalues problem associated with a $p \times p$ matrix, instead of the $n \times n$ Laplacian matrix, hence cutting cost from $O(n^3)$ flops to $O(p^3)$. A additional cost of the method lies in the building of the graph-Laplacian matrix, which requires $O(n^2p)$ flops, leading to a method scaling in $O(n^2p + p^3 + np^2)$ (the last $O(np^2)$ being due to the building of Φ). Beside being statistically inferior and computationally more expensive, note that graph-Laplacian also introduces an extra hyperparameter, which is the function (and its scale) to compute the weight matrix W.

Another drawback of graph-Laplacian is that its convergence to the real Laplacian is known up to a constant (Hein et al., 2007), which rescales all eigenvalues. To deal with this scaling constant, we assume $C = \sum_{i\in[k]}\lambda_i$ known for λ_i the true eigenvalues of $\mathcal{L}$ and $k = 25$ used to evaluate eigenvalues retrieval as per Figure 3, and we scale the graph Laplacian to ensure $\sum_{i\in[k]}\hat{\lambda}_i = C$. This can only improve the performance of graph-Laplacian, and only reinforce our findings on the superiority of Galerkin method, for which we do not employ this trick.

B.2 Additional Figures

To support empirically the claim made in Section 6.2, Figure 5 illustrates the learning of spherical harmonics with a neural network. Finally, Figure 6 shows Hermite polynomials in two dimensions, corresponding to the eigenfunctions of $\mathcal{L}_0$ with $\rho = \mathcal{N}(0, I)$, which could have served as a different basis to study our method. However, in high-dimension, $\mathcal{N}(0, I)$ tends to concentrate on the sphere and we do not expect many different behaviors in comparison to our study with spherical harmonics.

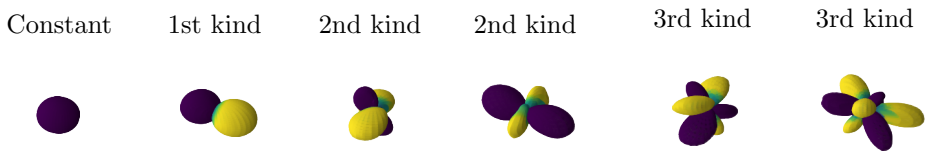

Figure 5: Cherry-picked learned spherical harmonics with a neural network. The network is a multi-layer perceptron with 200, 200, 2000, 200 hidden neurons in the four hidden layers, and $m = 16$ outputs optimized over 5000 batches of size 1000 with the contrastive version of the orthogonal regularizer. The optimizer is stochastic gradient descent with momentum ($m = 1/2$) initialized with a learning rate $\gamma = 10^{-3}$, with a scheduler to decrease the learning rate after one third and two third of the learning by a factor $1/3$. Principal component analysis was used to disentangle the learned representation and retrieve the different learned eigenspaces and eigenfunctions.

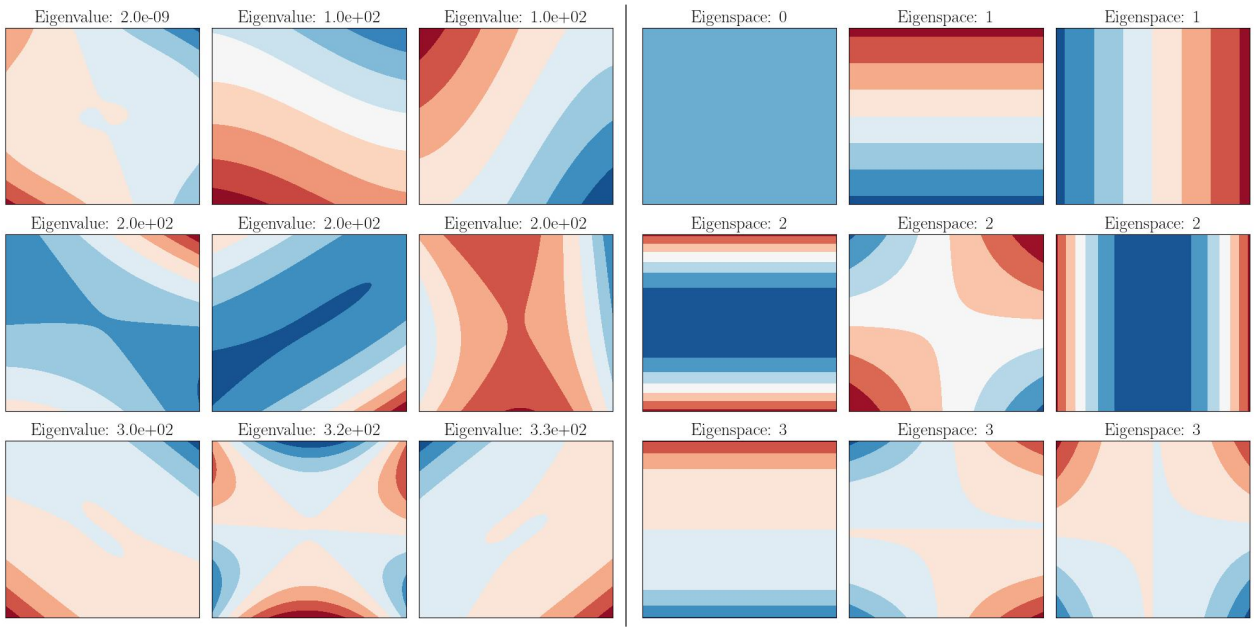

Figure 6: (Left) Level lines of the learned hermite polynomials in $2d$ thanks to the operator $\mathcal{L}$ with $\mathcal{X} = \mathbb{R}^2$ and $\rho = \mathcal{N}(0, I)$, with polynomial kernel of degree 3. Compared to the ground truth on the right, notice how the learned polynomials are "random" basis of the different eigenspace of $\mathcal{L}$. (Right) Canonical Hermite polynomials, acting as a ground truth for the left part. The title indicates the eigenspace number that the eigenfunctions belong to. Any orthogonal transformation $U(f_i)$ for (f_i) the k eigenfunctions of the k-th eigenspace, and U an orthogonal matrix in $\mathbb{R}^{k\times k}$ is a valid basis of eigenfunctions of this eigenspace, which is what is found in practice.

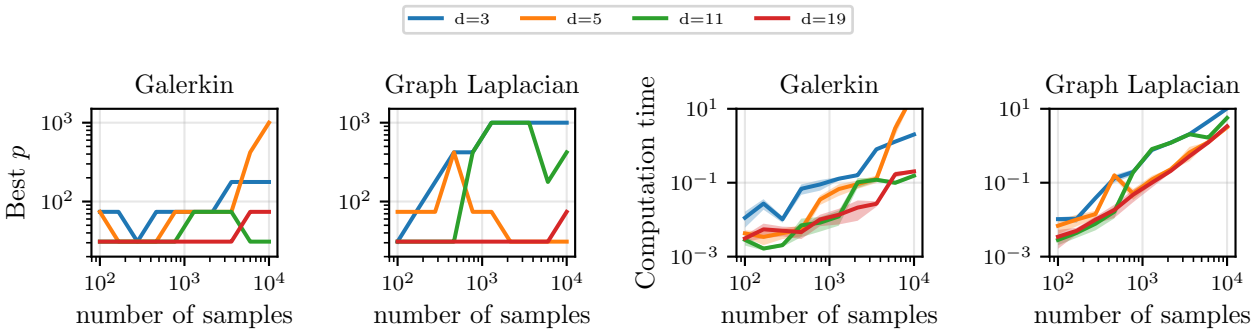

Figure 7: (Left) Value of p achieving the best results reported on Figure 3. (Right) Computation time corresponding to the best results.

B.3 Empirical Observations

While Figure 3 illustrates our take-home messages, i.e., "Galerkin beats graph-Laplacian, and its does not cost much in terms of implementation (scaling linearly with respect to n and being almost indifferent to the dimension)", much more could be said when digging into the one hundred runs with all the different hyperparameters.

Parameter grid. We consider the following values for the different experimental setups with spherical harmonics.

- n: ten values equally spaced in log space between 100 and 10000.
- $d \in \{3, 5, 7, \dots, 19\}$
- p five values equally spaced in log space between 30 and 1000.

We try three kernels for the Galerkin functions, together with five hyperparameters for each.

- The polynomial kernel $k_x(y) = (1 + x^\top y)^s$ with hyperparameter $s \in \{2, 3, 4, 5, 6\}$.
- The exponential kernel $k_x(y) = \exp(-\|x - y\| / \sigma)$ with hyperparameters $\sigma \in \{.1, 1, 10, 100, 1000\}$.
- The Gaussian kernel $k_x(y) = \exp(-\|x - y\|^2 / 2\sigma^2)$ with hyperparameter $\sigma \in \{.01, .1, 1, 10, 100\}$

For the Graph-Laplacian, we equally consider six different options for the weighting scheme in (16).

- Either $w_{ij} = k_{x_i}(x_j)$ with the same kernel as the Galerkin functions.
- Or $w_{ij} = \exp(-\|x_i - x_j)\|^2 / 2\sigma^2)$ with hyperparameters $\sigma \in \{.01, .1, 1, 10, 100\}$.

The result of Figure 3 were taken as the best over both p and each kernel and hyperparameters, leading to the best pick out of $5 \times 3 \times 5 = 75$ options for Galerkin, and out of $6 \times 75 = 450$ for graph-Laplacian. We ran one hundred trials for each configuration, and used Slurm to parallelize runs on a cluster of CPUs, together with the `SeedSequence` generator of Numpy to ensure a minimum correlation between pseudo-random runs (Harris et al., 2020).

Best results. First of all, we start by looking at the best results. The left of Figure 7 shows that the best values of p are not always the biggest ones. This is understandable, a bigger number of "Galerkin" functions leads to a bigger risk of overfitting, in particular in high-dimensions, which explains why the red curves are below the other ones. The right of Figure 7 showcases the corresponding computation time, they are highly correlated with the value of p, and they are of similar order for graph-Laplacian and the Galerkin method. Finally, Figure 8 makes sure that the best hyperparameter values were not obtained for extreme values of our parameter grid, removing confounders due to bad hyperparameters calibration.

Effect of the different parameters at play. Figure 9 explores the effect of the dimension on our estimator quality. Figure 10 explores the effect of the kernel used for the Galerkin method. We notice that although the eigenfunctions are polynomials, the polynomial kernel does not necessarily lead to the best result, this is due to the fact that there are many polynomials in dimension d, and the polynomial kernel does favor the learning of simple polynomials over more complex ones. We again notice the regularizing effect of not choosing the biggest p possible (e.g., choosing $p = 1000$ as soon as $n \geq 1291$ according to our grid values).

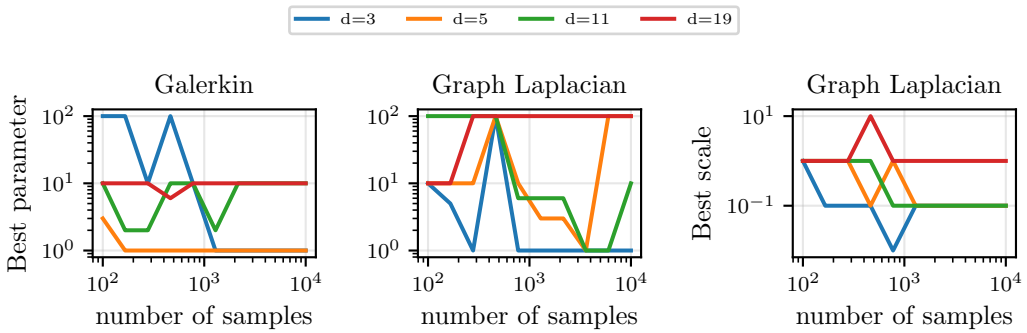

Figure 8: (Left) Best parameters for Galerkin functions, they are strictly inside our grid of parameters (for graph-Laplacian, the best results are obtained with the exponential kernel for which our grid stops at 10^3). (Right) Best parameters for the weights w_{ij} in (16). Once again, the best values are obtained strictly inside the grid.

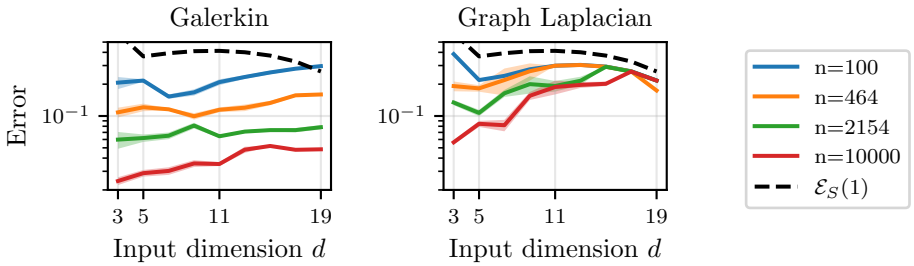

Figure 9: *Effect of the dimension.* Scaling of the error (21) as a function of the input dimension d for the best set of hyperparameters for both the Galerkin method and graph-Laplacian. Note that the problem changes as the dimension augments, and that we are rescaling the error so that $\mathcal{E}_S(0) = 1$ regardless of the dimension. The graph-Laplacian seems to reach a default error as d increases which should be put in comparison with the error reached by $\hat{\lambda}_i = 1/\sum_{i\in[k]} \lambda_i$ plotted in black (recall that we artificially rescaled the graph-Laplacian to ensure $\sum_{i\in[k]} \hat{\lambda}_i = \sum_{i\in[k]} \lambda_i$).

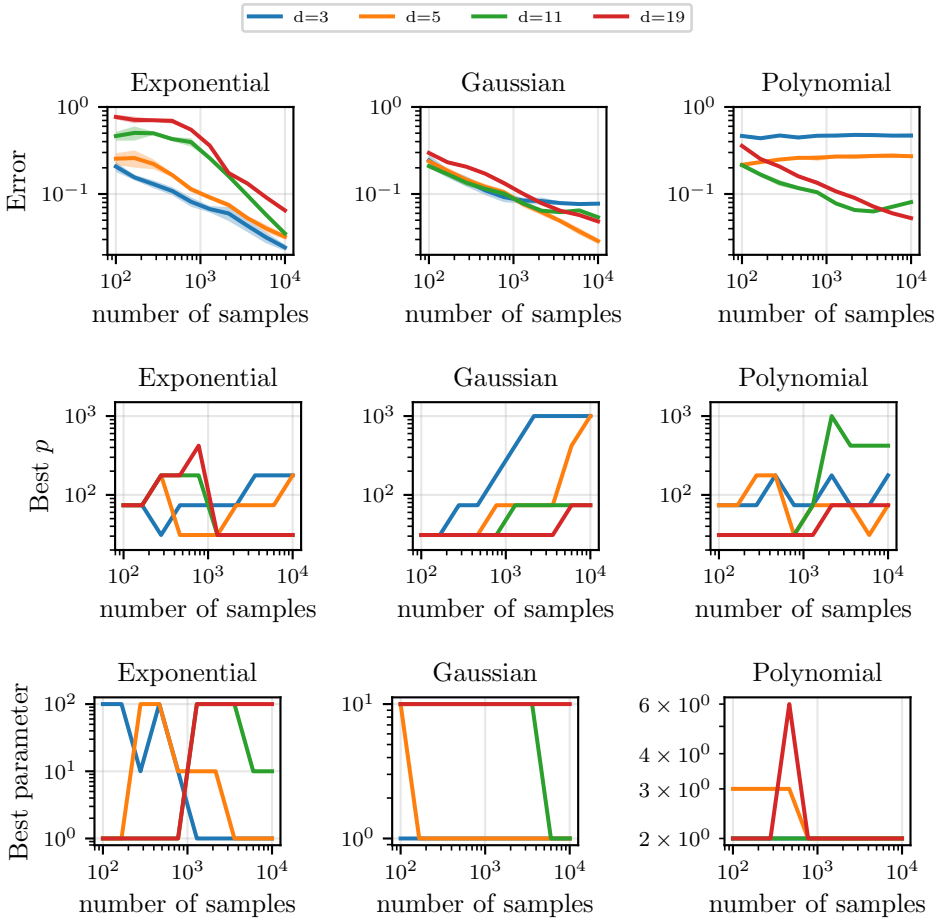

Figure 10: Influence on the kernel for Galerkin method.

B.4 Hermite Regression

Consider the Hermite interpolation problem, where for $(x_i)_{i\in[n]}$ n data point in $\mathcal{X} = \mathbb{R}^d$, (v_i) n scalar values, and (t_i) n vectors in $\mathbb{R}^d$, we try to solve

$$\arg\min_{f\in\mathcal{F}} \sum_{k\in[n]} \|f(x_k) - v_k\|^2 + \|\nabla f(x_k) - t_k\|^2 ,$$

for some space of functions $\mathcal{F}$. In our case, we consider

$$\mathcal{F} = \operatorname{Span}\{\varphi_i \,|\, i \in [p]\} ,$$

which will not ensure interpolation when $p < n(d+1)$, hence our denomination of "Hermite regression". With $\alpha \in \mathbb{R}^p$ and $f = F\alpha = \sum_{i\in[n]} \alpha_i\varphi_i$, this leads to

$$\begin{aligned}
&\arg\min_{\alpha\in\mathbb{R}^p} \sum_{k\in[n]} \Big\| \sum_{i\in[p]} \alpha_i\varphi_i(x_k) - v_k\Big\|^2 + \Big\| \sum_{i\in[p]} \alpha_i\nabla\varphi_i(x_k) - t_k\Big\|^2 \\
&= \arg\min_{\alpha\in\mathbb{R}^p} \sum_{k\in[n]} \sum_{i,j\in[p]} \alpha_i\alpha_j \left(\varphi_i(x_k)\varphi_j(x_k) + \sum_{l\in[d]} \partial_l\varphi_i(x_k)\partial_l\varphi_j(x_k)\right) - 2\sum_{i\in[p]} \alpha_i \left(\varphi_i(x_k)v_k + \sum_{l\in[d]} \partial_l\varphi_i(x_k)t_{kl}\right) \\
&= \arg\min_{\alpha\in\mathbb{R}^p} \alpha^\top A\alpha - 2b^\top\alpha = A^{-1}b,
\end{aligned}$$

where $A \in \mathbb{R}^{p\times p}$ and $b \in \mathbb{R}^p$ are defined as

$$A_ij = \sum_{k\in[n]} \varphi_i(x_k)\varphi_j(x_k) + \langle\nabla\varphi_i(x_k), \nabla\varphi_j(x_k)\rangle , \qquad b_i = \sum_{k\in[n]} \varphi_i(x_k)v_k + \langle\nabla\varphi_i(x_k), t_k\rangle .$$

A naive implementation leads to $O(np^2d + npd)$ flops and $O(p^2)$ bits to build those matrices, and $O(p^3)$ to solve $\alpha = A^{-1}b$. In the case where $\varphi_i = k_{x_i}$ with k a dot-product or an invariant kernel, Proposition 1 shows how to reduce the time complexity to $O(np^2 + npd)$ flops at the expense of a memory complexity in $O(p^2 + np)$ bits. Note that similarly to Proposition 1 the term b can also be written in a simple form for structured kernels, although this does not reduce the overall complexity of building b.

Proposition 10. *Assume that $\mathcal{X}$ is endowed with a scalar product. Given a kernel $k_x(y) = q(\|x-y\|)$ defined from $q : \mathbb{R} \to \mathbb{R}$, for $x, y, t \in \mathcal{X}$, it holds*

$$\langle\nabla k_y(x), t\rangle = \frac{q'(\|x-y\|)}{\|x-y\|} (x-y)^\top t. \tag{23}$$

Similarly for dot-product kernel $k_x(y) = q(x^\top y)$,

$$\langle\nabla k_y(x), t\rangle = q'(x^\top y)\, y^\top t. \tag{24}$$

Proof. Once again, the proof follows from the application of the chain rule. □

Propositions 1 and 10 explain our Hermite regression algorithms with dot-product kernel, Algorithm 4, and translation-invariant kernel, Algorithm 5.

Algorithm 4: Hermite regression with dot-product kernel

Data: Data $(x_i) \in \mathbb{R}^{n\times p}$, $V = (v_i) \in \mathbb{R}^n$, $(t_i) \in \mathbb{R}^{n\times p}$, kernel $k_x(y) = q(x^\top y)$.
Compute $X = (x_i^\top x_j) \in \mathbb{R}^{p\times n}$ with $p \leq n$;
Compute $\Psi = q(X)q(X)^\top \in \mathbb{R}^{p\times p}$ elementwise;
Compute $L = (q'(X)q'(X)^\top)$;
Update $L_{ij} \leftarrow X_{ij}L_{ij}$ for all $i, j \in [p]$;
Set $A = L + \Psi$;
Compute $T = (x_i^\top t_j) \in \mathbb{R}^{p\times n}$;
Update $T \leftarrow q'(X) \odot T$ with elementwise (Hadamard) product;
Compute $b = q(X)V + T1_n \in \mathbb{R}^p$ where $1_n = (1)_{i\in[n]} \in \mathbb{R}^n$;
Solve $\alpha = A^{-1}b$;
Set $\hat{f}(x) := \sum_{i\in[p]} \alpha_i k_{x_i}(x)$.
Result: Hermite estimator $\hat{f}$

Algorithm 5: Hermite regression with distance kernel

Data: Data $(x_i) \in \mathbb{R}^{n\times p}$, $V = (v_i) \in \mathbb{R}^n$, $(t_i) \in \mathbb{R}^{n\times p}$, kernel $k_x(y) = q(\|x - y\|)$.
Compute $X = (x_i^\top x_j), \in \mathbb{R}^{p\times n}$, $D = (x_i^\top x_i) \in \mathbb{R}^n$;
Deduce $N = (\|x_i - x_j\|) \in \mathbb{R}^{p\times n}$ and $T = q'(N)/N$;
Initialize $L = 0 \in \mathbb{R}^{p\times p}$; $\Psi = q(N)q(N)^\top \in \mathbb{R}^{p\times p}$;
for $k \in [n]$ **do**
  Set $\gamma_{ij}^{(k)} := (D_k - X_{ik} - X_{jk} + X_{ij})$;
  Update $L_{ij} \leftarrow L_{ij} + \gamma_{ij}^{(k)} T_{ik}T_{jk}$;
Set $A = L + \Psi$;
Compute $T = -(x_i^\top t_j) \in \mathbb{R}^{p\times n}$;
Update $T_{ij} \leftarrow T_{ij} + x_j^\top t_j$ for all $i \in [p]$, $j \in [n]$;
Update $T \leftarrow T \odot (q'(N)/N)$ with elementwise (Hadamard) product;
Compute $b = q(N)V + T1_n \in \mathbb{R}^p$ where $1_n = (1)_{i\in[n]} \in \mathbb{R}^n$;
Solve $\alpha = A^{-1}b$;
Set $\hat{f}(x) := \sum_{i\in[p]} \alpha_i k_{x_i}(x)$.
Result: Hermite estimator $\hat{f}$