Enhanced Feature Learning via Regularisation: Integrating Neural Networks and Kernel Methods

Bertille Follain BERTILLE.FOLLAIN@INRIA.FR
Inria, Département d'Informatique de l'Ecole Normale Supérieure, PSL Research University
48 rue Barrault, 75013, Paris, France

Francis Bach FRANCIS.BACH@INRIA.FR
Inria, Département d'Informatique de l'Ecole Normale Supérieure, PSL Research University
48 rue Barrault, 75013, Paris, France

Abstract

We propose a new method for feature learning and function estimation in supervised learning via regularised empirical risk minimisation. Our approach considers functions as expectations of Sobolev functions over all possible one-dimensional projections of the data. This framework is similar to kernel ridge regression, where the kernel is $\mathbb{E}_w(k^{(B)}(w^\top x, w^\top x'))$, with $k^{(B)}(a,b) := \min(|a|,|b|)\mathbb{1}_{ab>0}$ the Brownian kernel, and the distribution of the projections w is learnt. This can also be viewed as an infinite-width one-hidden layer neural network, optimising the first layer's weights through gradient descent and explicitly adjusting the non-linearity and weights of the second layer. We introduce an efficient computation method for the estimator, called Brownian Kernel Neural Network (BKerNN), using particles to approximate the expectation. The optimisation is principled due to the positive homogeneity of the Brownian kernel. Using Rademacher complexity, we show that BKerNN's expected risk converges to the minimal risk with explicit high-probability rates of $O(\min((d/n)^{1/2}, n^{-1/6}))$ (up to logarithmic factors). Numerical experiments confirm our optimisation intuitions, and BKerNN outperforms kernel ridge regression, and favourably compares to a one-hidden layer neural network with ReLU activations in various settings and real data sets.

Keywords: feature learning, neural network, reproducing kernel Hilbert space, regularised empirical risk minimisation, Rademacher complexity

1 Introduction

In the era of high-dimensional data, effective feature selection methods are crucial. Representation learning aims to automate this process, extracting meaningful information from complex data sets. Non-parametric methods often struggle in high-dimensional settings, making the multi-index model, which assumes a few relevant linear features explain the relationship between response and factors, an attractive alternative. Formally, the multiple index model (Xia, 2008) is expressed as $Y = f^*(X) + \text{noise} = g^*(P^\top X) + \text{noise}$, with Y the response, X the d-dimensional covariates, g^* the unknown link function, $P \in \mathbb{R}^{d\times k}$ the features and $k \leq d$, the number of such relevant linear features. The components $P^\top X$ are linear features of the data that need to be learnt, reducing the dimensionality of the problem, which may allow to escape the curse of dimensionality, while the more general function g increases the capacity of the model.

Multiple index models have been extensively studied, leading to various methods for estimating the feature space. Brillinger (2012) introduced the method of moments for Gaussian data and one feature, by using specific moments to eliminate the unknown function. For features of any dimension, several methods have been proposed. Sliced inverse regression (SIR) (Li, 1991) uses second-order moments to identify effective dimensions by slicing the response variable and finding linear combinations of predictors, while improvements have been proposed (Yang et al., 2017), these methods heavily rely on assumptions about the covariate distribution shape and prior knowledge of the distribution. Iterative improvements have been an interesting line of work (Dalalyan et al., 2008), while optimisation-based methods like local averaging minimise an objective function to estimate the subspace (Fukumizu et al., 2009; Xia et al., 2002). Despite their practical performance, particularly the MAVE method (Xia et al., 2002), the theoretical guarantees show exponential dependence on the original data dimension, making them less suitable for high-dimensional settings.

In this work, we tackle feature learning and function estimation jointly through the paradigm of empirical risk minimisation. We consider a classical supervised learning problem. We have i.i.d. samples $(x_i, y_i)_{i\in[n]}$ from a random variable $(X, Y) \in \mathcal{X} \times \mathcal{Y} \subset \mathbb{R}^d \times \mathbb{R}$. Our goal is to minimise the expected risk, which is defined as $\mathcal{R}(f) := \mathbb{E}_{X,Y}[\ell(Y, f(X))]$ over some class of functions $\mathcal{F}$, where ℓ is a loss function mapping from $\mathbb{R} \times \mathbb{R}$ to $\mathbb{R}$. This can be achieved through the framework of regularised empirical risk minimisation, where the empirical risk is defined as $\widehat{\mathcal{R}}(f) := \frac{1}{n}\sum_{i=1}^n \ell(y_i, f(x_i))$. Our interest in regularised empirical risk minimisation stems from its flexibility, allowing it to be applied to a wide range of problems as long as the objective can be defined as the optimisation of an expected loss. Our primary objective is to achieve the lowest possible risk, which we study in theory and in practice, while we explore the recovery of underlying features in numerical experiments. Our method draws inspiration from several lines of work, namely positive definite kernels and neural networks with their mean field limit, which we briefly review, together with the main limitations we aim to alleviate.

Kernel methods and multiple kernel learning. A well-known method in supervised learning is kernel ridge regression (KRR, Vovk, 2013), which implicitly maps data into high-dimensional feature spaces using kernels. It benefits from dimension-independent rates of convergence if the model is well-specified, i.e., if the target function belongs to the related Hilbert space. However, KRR does not benefit from the existence of linear features in terms of convergence rates of the risk when the model is misspecified (Bach, 2024, Section 9.3.5), as it relies on pre-specified features. To address the limitations of single-kernel methods, multiple kernel learning (MKL) optimally combines multiple kernels to capture different data aspects (Bach et al., 2004; Gönen and Alpaydın, 2011). However, MKL suffers from significant computational complexity and the critical choice of base kernels, which can introduce biases if not selected properly. Furthermore, MKL does not resolve the issue of leveraging hidden linear features effectively.

Neural networks. Now consider another type of supervised learning methods, namely neural networks with an input layer of size d, a hidden layer with m neurons, an activation function σ, followed by an output layer of size 1. Functions which can be represented are of the form $f(x) = \sum_{j=1}^m \eta_j \sigma(w_j^\top x + b_j)$, where σ can be the ReLU, $\sigma(z) = \max(0, z)$ or the step function $\sigma(z) = \mathbb{1}_{z>0}$. Neural networks benefit from hidden linear features, achieving

favourable rates dependent on k, the number of relevant features, rather than on the data dimension (Bach, 2024, Section 9.3.5). However, this formulation requires multiple b values to fit a function with the same w, particularly in single-index models $f^*(x) = g^*(w^\top x)$, which is inefficient.

Regularising by adding a penalty term to the empirical risk minimisation objective guides specific estimator behaviours. In the context of feature learning, Rosasco et al. (2013) used derivatives for regularisation in nonparametric models, focusing on variable selection. While their method reduces to classical regularisation techniques for linear functions, it faces limitations: functions depending on a single variable do not belong in the chosen RKHS, using derivatives at data points limits the exploitation of regularity, and there is no benefit from hidden variables in the misspecified case. An improvement over this method was studied for both feature learning and variable selection by Follain and Bach (2024), where a trace norm penalty (Koltchinskii et al., 2011; Giraud, 2014) on the derivatives was used for the feature learning case. However, the dependency on the dimension of the rate did not allow high-dimensional learning. We can justify the use of trace norm penalties by considering the structure of neural networks. Under the multiple-index model, the weights $w_1, \dots, w_m$ of the first layer are expected to lie in a low-rank subspace of rank at most k. However, directly enforcing a rank constraint is not practical for optimisation. Therefore, we could use a relaxation such as $\Omega(f) = \operatorname{tr}\left(\left(\sum_{j=1}^m w_j w_j^\top\right)^{1/2}\right)$, which is the trace norm of a matrix containing the weights, to approximate the rank constraint effectively. However, there is still the issue of multiple constant terms for a single weight. We will see specialised penalties for feature learning for a different family of functions.

Mean-field limit. To apply a similar framework to our future estimator, we introduce the mean-field limit of an over-parameterised one-hidden layer neural network (Nitanda and Suzuki, 2017; Mei et al., 2019; Chizat and Bach, 2022; Sirignano and Spiliopoulos, 2020). When the number of neurons m is very large, the network can be rescaled as follows

$$f(x) = \frac{1}{m}\sum_{j=1}^m \eta_j \sigma(w_j^\top x + b_j), \text{ which approximates } \int \eta \sigma(w^\top x + b)\,\mathrm{d}\mu(\eta, w, b), \qquad (1)$$

where μ is a probability distribution, and we can take the weights and constant terms (w, b) to be constrained when the activation is 1-homogeneous,[1] such as the ReLU or step function. This approach is valuable because, as noted by Chizat and Bach (2022), under certain conditions (convexity of the loss and penalty functions, homogeneity of the activation function), the regularised empirical risk problem optimised via gradient descent in the infinitely small step-size limit converges to the minimiser of the corresponding problem with infinitely many particles. This allows us to use a finite number of particles m in practice while still leveraging the theoretical benefits derived from the continuous framework.

1.1 Plan of the Paper and Notations

In this paper, we introduce the Brownian kernel neural network (BKerNN), a novel model for feature learning and function estimation. Our approach combines kernel methods and

1. A function Φ is positively 1-homogeneous if, for any $\kappa > 0$, $\Phi(\kappa w) = \kappa \Phi(w)$.

neural networks using regularised empirical risk minimisation. Section 2 presents the theoretical foundations and formulation of BKerNN. Section 3 details the practical implementation, including the optimisation algorithm and convergence insights. Section 4 provides a statistical analysis using Rademacher complexity to show high-probability convergence to the minimal risk with explicit rates. Section 5 evaluates BKerNN through experiments on simulated and real data sets, comparing it with neural networks and kernel methods. Finally, Section 6 summarises the findings and suggests future research directions.

We use the following notations. For a positive integer m, we define $[m] := \{1, \dots, m\}$. For a d-dimensional vector α and $i \in [d]$, α_i denotes its i-th element. For a matrix A, $\operatorname{tr} A$ denotes its trace when A is square, A^{-1} its inverse when well defined, while $A_{i,j}$ the element in its i-th row and j-th column, and $A^\top$ its transpose. I_d is the $d \times d$ identity matrix. We use $\mathcal{S}^{d-1}$ to denote the unit sphere in $\mathbb{R}^d$ for $\|\cdot\|$ a generic norm and $\|\cdot\|_*$ its dual norm. The ℓ_2, ℓ_1, and ℓ_∞ norms are denoted as $\|\cdot\|_2$, $\|\cdot\|_1$, and $\|\cdot\|_\infty$ respectively. We use $O(\cdot)$ to denote the asymptotic behaviour of functions, indicating the order of growth. The set of probability measures on a given space S is denoted by $\mathcal{P}(S)$. A normal random variable is denoted as following the law $\mathcal{N}(\text{mean}, \text{variance})$. $\mathbb{1}$ is the indicator function. For two spaces S_1, S_2, $S_1^{S_2}$ is the set of functions from S_2 to S_1.

2 Neural Networks and Kernel Methods Fusion

Building on the limitations of current methods discussed in the introduction, we propose a novel architecture that integrates neural networks with kernel methods. This approach can be interpreted in two ways: as learning with a kernel that is itself learned during training, or as employing a one-hidden layer neural network where the weights from the input layer to the hidden layer are learned through gradient descent, while the weights and non-linearity from the hidden layer to the output are optimised explicitly. In this section, we introduce the custom function space we propose, revisit key properties of reproducing kernel Hilbert spaces (RKHS), and explore the connections between BKerNN model, kernel methods, and neural networks. Additionally, we present the various regularisation penalties we consider throughout our analysis.

2.1 Custom Space of Functions

We begin by considering the continuous setting, which mirrors the mean-field limit of over-parameterised one-hidden layer neural networks discussed in Section 1.

Definition 1 (Infinite-Width Function Space)) *Let*

$$\mathcal{F}_\infty := \left\{ f \mid f(\cdot) = c + \int_{\mathcal{S}^{d-1}} g_w(w^\top \cdot)\, \mathrm{d}\mu(w) \right\},$$

where c is a constant in $\mathbb{R}$, $\mathcal{S}^{d-1}$ is the unit sphere for some norm $\|\cdot\|$ on $\mathbb{R}^d$ (typically either ℓ_2 or ℓ_1), $\mu \in \mathcal{P}(\mathcal{S}^{d-1})$ is a probability measure on $\mathcal{S}^{d-1}$, and $\forall w \in \mathcal{S}^{d-1}$, $g_w : \mathbb{R} \to \mathbb{R}$ belongs to a space of functions $\mathcal{H}$. We define $\mathcal{H}$ as $\{g : \mathbb{R} \to \mathbb{R} \mid g(0) = 0,\ g$ has a weak derivative $g',\ \int_{\mathbb{R}} (g')^2 < \infty\}$. $\mathcal{H}$ is a Hilbert space and a Sobolev space, with the inner product defined as $\langle \tilde{g}, g \rangle_{\mathcal{H}} = \int \tilde{g}' g'$. Note that g_w vary for each w.

The function space $\mathcal{F}_\infty$ is inspired by infinite-width single hidden layer neural networks: with the addition of the intercept c, each function in this space can be seen as the integral of a linear part w and a non-linearity g_w over some probability distribution, as in Equation (1) where the non-linearity is $\eta\sigma(\cdot)$. Thus here the activation functions are learnt.

The approximation of $\mathcal{F}_\infty$ with m particles can then be obtained as follows.

Definition 2 (Finite-Width Function Space) *Let*

$$\mathcal{F}_m := \left\{ f \mid f(\cdot) = c + \frac{1}{m}\sum_{j=1}^{m} g_j(w_j^\top \cdot), w_j \in \mathcal{S}^{d-1}, g_j \in \mathcal{H}, c \in \mathbb{R} \right\}.$$

Remark that $\forall m \in \mathbb{N}^*, \mathcal{F}_m \subset \mathcal{F}_\infty$, by taking the discrete probability measure uniformly supported by the particles $w_1, \dots, w_m$.

We now consider regularised empirical risk minimisation starting with a basic penalty. Let

$$\Omega_0(f) = \inf_{c\in\mathbb{R},(g_w)_w\in\mathcal{H}^{\mathcal{S}^{d-1}},\mu\in\mathcal{P}(\mathcal{S}^{d-1})} \int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}} \,\mathrm{d}\mu(w), \tag{2}$$

such that $f = c + \int_{\mathcal{S}^{d-1}} g_w(w^\top \cdot)\,\mathrm{d}\mu(w)$, where $\|g_w\|_{\mathcal{H}}^2 := \int_{-\infty}^{+\infty} g_w'(t)^2\,\mathrm{d}t$. This penalty enforces the regularity of the function and, because we use penalizations with non-squared norms, limits the number of non-zero g_w. While this penalty is not specifically aimed at feature learning, by limiting the number of non-zero particles, it indirectly promotes feature learning to some extent. This serves as a starting point, and we introduce more targeted penalties in Section 2.5 with a stronger feature learning behavior. For $f \in \mathcal{F}_m$ written as in Definition 2, the penalty simplifies to $\Omega_0(f) = \frac{1}{m}\sum_{j=1}^m \|g_j\|_{\mathcal{H}}$. The learning objective is thus defined as

$$\hat{f}_\lambda := \operatorname*{arg\,min}_{f\in\mathcal{F}} \ \widehat{\mathcal{R}}(f) + \lambda\Omega(f), \tag{3}$$

where $\lambda > 0$ is a regularisation parameter and Ω is currently Ω_0 from Equation (2). The function space $\mathcal{F}$ is either $\mathcal{F}_\infty$ or $\mathcal{F}_m$. For statistical analysis in Section 4, we consider $\mathcal{F}_\infty$, while in practice, we compute the estimator using $\mathcal{F}_m$ as discussed in Section 3. The rationale for using $\mathcal{F}_m$ and expecting the statistical properties of $\mathcal{F}_\infty$ is elaborated in Section 3.2.

In the continuous setting, Equation (3) corresponds to

$$\min_{c\in\mathbb{R},(g_w)_w\in\mathcal{H}^{\mathcal{S}^{d-1}},\mu\in\mathcal{P}(\mathcal{S}^{d-1})} \frac{1}{n}\sum_{i=1}^{n} \ell\left(y_i, c + \int_{\mathcal{S}^{d-1}} g_w(w^\top x_i)\mathrm{d}\mu(w)\right) + \lambda \int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}}\mathrm{d}\mu(w), \tag{4}$$

while in the m-particles setting, Equation (3) becomes

$$\min_{c\in\mathbb{R},w_1,\dots,w_m\in\mathcal{S}^{d-1},g_1,\dots,g_m\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \ell\left(y_i, c + \frac{1}{m}\sum_{j=1}^{m} g_j(w_j^\top x_i)\right) + \lambda\frac{1}{m}\sum_{j=1}^{m} \|g_j\|_{\mathcal{H}}. \tag{5}$$

where we clearly see a sparsity-inducing "grouped" penalty (Yuan and Lin, 2006).

2.2 Properties of Reproducing Kernel Hilbert Space $\mathcal{H}$ and Kernel k

In this subsection, we succinctly present some properties of reproducing kernel Hilbert spaces (RKHS) that are essential for our analysis. See Aronszajn (1950); Berlinet and Thomas-Agnan (2011) for an introduction to RKHS. Recall that we defined the Hilbert space $\mathcal{H}$ as

$$\mathcal{H} := \left\{ g : \mathbb{R} \to \mathbb{R} \mid g(0) = 0, \int_{\mathbb{R}} (g')^2 < +\infty \right\},$$

with the inner product $\langle \tilde{g}, g \rangle = \int_{\mathbb{R}} \tilde{g}' g'$. This space is a reproducing kernel Hilbert space with the reproducing kernel $k^{(B)}(a,b) = (|a| + |b| - |a-b|)/2 = \min(|a|,|b|)\mathbb{1}_{ab>0}$. This kernel, which can be referred to as the "Brownian" kernel, corresponds to the covariance of the Brownian motion at times a and b (Mishura and Shevchenko, 2017). Consequently, we have the reproducing property

$$\forall g \in \mathcal{H}, \forall a \in \mathbb{R}, g(a) = \langle g, k_a^{(B)} \rangle,$$

where $k_a^{(B)} : b \in \mathbb{R} \to k^{(B)}(a,b) \in \mathbb{R}$. As a reproducing kernel, it is positive definite, meaning that for any $n \in \mathbb{N}$, $\alpha \in \mathbb{R}^n$, and $a \in \mathbb{R}^n$, we have $\sum_{i,j=1}^n \alpha_i k^{(B)}(a_i, a_j)\alpha_j \geq 0$. Additionally, we observe that $\|k_a^{(B)}\|_{\mathcal{H}}^2 = |a|$ and $\|k_a^{(B)} - k_b^{(B)}\|_{\mathcal{H}}^2 = |a-b|$. It is also noteworthy that by definition, the functions in $\mathcal{H}$ are necessarily continuous, in fact even 1/2-Hölder continuous as we see in Lemma 3.

The usual Hilbert/Sobolev space is $W^{1,2}(\mathbb{R})$ (also written as $\mathcal{H}^1$) with inner product equal to $\langle f, g \rangle = \int fg + \int f'g'$. This space is also an RKHS for the reproducing kernel $k^{\exp}(a,b) = \exp(-|a-b|)$ (see, e.g., Williams and Rasmussen, 2006). We demonstrate that for optimisation purposes, the Brownian kernel is more advantageous due to its positive homogeneity in Section 3.2.

2.3 Characterisation of $\mathcal{F}_\infty$

In this subsection, we discuss the properties of the function space $\mathcal{F}_\infty$ and its relationship to other relevant spaces, such as the space of functions of one-hidden-layer neural networks presented in Section 1. We first present the following lemma.

Lemma 3 (Properties of Functions in $\mathcal{F}_\infty$) *$\mathcal{F}_\infty$ is a vector space, $\max(f(0), \Omega_0(f))$ is a norm on $\{f \in \mathcal{F}_\infty \mid \Omega_0(f) < \infty\}$. For $f \in \mathcal{F}_\infty$ with $\Omega_0(f) < \infty$, the function f is 1/2-Hölder continuous with constant $\Omega_0(f)$, i.e., $|f(x) - f(x')| \leq \Omega_0(f)\sqrt{\|x - x'\|^*}$. Additionally, f has weak derivatives belonging to $L_2(\rho)$ for any probability measure ρ on $\mathbb{R}^d$.*

The proof can be found in Appendix A.1.1. This lemma indicates that the space of functions $\mathcal{F}_\infty$ is contained within the space of 1/2-Hölder continuous functions with weak derivatives that are square integrable with respect to any probability measure. Recall that on a compact, all Lipschitz functions are Hölder continuous functions, indicating that the Hölder condition is less restrictive.

Now, we consider the relationship of $\mathcal{F}_\infty$ to other function spaces. Starting with the one-dimensional case, BKerNN reduces to kernel ridge regression with the Brownian kernel, which is also equivalent to learning with natural cubic splines (for an introduction to splines, see Wahba, 1990). For multi-dimensional data, we use the Fourier decomposition of functions to bound the defining norms of function spaces, enabling us to make comparisons.

Lemma 4 (Functions Spaces Included in $\mathcal{F}_\infty$) *Assume we only consider functions f with support on the ball centred at 0 with radius R and norm $\|\cdot\|^*$. Assume f has a Fourier decomposition,*

$$f(x) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{f}(\omega) e^{i\omega^\top x} \,\mathrm{d}\omega,$$

then, it follows that

$$\Omega_0(f) \leq \frac{\sqrt{2R}}{(2\pi)^d} \int_{\mathbb{R}^d} |\hat{f}(\omega)| \cdot \|\omega\| \,\mathrm{d}\omega.$$

Hence, if $\int_{\mathbb{R}^d} |\hat{f}(\omega)| \cdot \|\omega\| \,\mathrm{d}\omega < \infty$, then f belongs to $\mathcal{F}_\infty$.

The proof is given in Appendix A.1.2. We remark that the condition $\int_{\mathbb{R}^d} |\hat{f}(\omega)| \cdot \|\omega\| \,\mathrm{d}\omega < \infty$ is a form of constraint on the regularity of the first-order derivatives.

According to Bach (2024, Section 9.3.4), the space of one-hidden-layer neural networks with ReLU activations in the mean-field limit with $\|w\|_2 = 1, |b| \leq R$ can be equipped with the Banach norm $\gamma_1(f) = \int |\eta| \mathrm{d}\mu(\eta, w, b)$, which can be then bounded as in Lemma 3 by

$$\frac{2}{(2\pi)^d R} \int_{\mathbb{R}^d} |\hat{f}(\omega)| (1 + 2R^2 \|\omega\|_2^2) \,\mathrm{d}\omega. \tag{6}$$

Now remark that the bound on Ω_0 contains a factor $\|w\|$ in the integral, whereas for ReLU neural networks with γ_1 norm it is $1 + 2R^2\|w\|_2^2$. Hence, the constraint is stronger on the neural network space space, no matter what norm $\|\cdot\|$ corresponds to, suggesting that $\mathcal{F}_\infty$ is a larger space of functions.

Also note that the bound from Equation (6) can be shown to be smaller (up to a constant) than the norm defining the Sobolev space penalising derivatives up to order $s := d/2 + 5/2$, which is $\int_{\mathbb{R}^d} |\hat{f}(\omega)|^2 (1 + 2R^2\|\omega\|_2^2)^s \,\mathrm{d}\omega$. (Bach, 2024, Section 9.3.5). This space is an RKHS because $s > d/2$, and the inequality on norms yields that the space of neural networks with ReLU activations equipped with the norm γ_1 (which is a Banach space) contains this RKHS. Another interesting remark is that if we used the norm $\gamma_2(f) = \int \eta^2 \mathrm{d}\mu(\eta, w, b)$ instead of γ_1, the space that we would obtain is an RKHS and is strictly included in the one defined by γ_1 (Bach, 2024, Section 9.5.1)

For neural networks with step activations, i.e., $\sigma(z) = \mathbb{1}_{z>0}$ in the mean-field limit, a similar bound holds for the γ_1 norm

$$\frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} |\hat{f}(\omega)| (1 + R\|\omega\|_2) \,\mathrm{d}\omega. \tag{7}$$

This can be seen by applying the same proof technique as for Equation (6) from Bach (2024, Section 9.3.4)[2]. Learning with this space is not practically feasible due to optimisation issues as the step function is incompatible with gradient descent methods. However, the bound from Equation (7) is similar to the one on Ω_0, hinting that $\mathcal{F}_\infty$ is comparably large even though learning is possible with $\mathcal{F}_\infty$, as discussed in Section 3. For a discussion on this topic, see Bach (2024, Chapter 9) and Liu et al. (2024).

2. The only difference being that we use $e^{iu\|w\|_2} = 1 + \int_0^R i\|w\|_2 e^{it\|w\|_2} \mathbb{1}_{t \leq u} \mathrm{d}t$ instead of Taylor's formula, yielding $\gamma_1(x \to e^{i\omega^\top x}) \leq 1 + R\|w\|_2$.

2.4 Learning the Kernel or Training a Neural Network?

We first transform the optimisation problem before considering our setup from two different perspectives: one through kernel learning and the other through neural networks. To transform the optimisation problem, we use the representer theorem, a well-known result in RKHS that allows us to replace the optimisation over functions in the RKHS with optimisation over a finite weighted sum of the kernel at the data points.

Lemma 5 (Kernel Formulation of Finite-Width) *Equation* (5) *is equivalent to*

$$\min_{w_1,\dots,w_m\in\mathbb{R}^d, c\in\mathbb{R}, \alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^n \ell(y_i,(K\alpha)_i+c) + \frac{\lambda}{2}\alpha^\top K\alpha + \frac{\lambda}{2}\frac{1}{m}\sum_{j=1}^m \|w_j\|, \tag{8}$$

where $K = \frac{1}{m}\sum_{j=1}^m K^{(w_j)}$, *and* $K^{(w_j)} \in \mathbb{R}^{n\times n}$ *is the kernel matrix for kernel* $k^{(B)}$ *and projected data* $(w_j^\top x_1,\dots,w_j^\top x_n)$, *i.e.,* $K^{(w_j)}_{i,i'} = (|w_j^\top x_i| + |w_j^\top x_{i'}| - |w_j^\top(x_i - x_{i'})|)/2$. *Notice that there are no constraints on the particles* $(w_j)_{j\in[m]}$ *to belong to the unit sphere anymore.*

The proof is provided in Appendix A.2.1. This lemma shows that we only need to solve a problem over finite-dimensional quantities. For computational complexity considerations, see Section 3.1. We can view Equation (8) using kernels. In a classical kernel supervised learning problem with an unregularised intercept, we would have a fixed kernel matrix K and consider

$$\min_{c\in\mathbb{R},\alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^n \ell(y_i,(K\alpha)_i+c) + \frac{\lambda}{2}\alpha^\top K\alpha.$$

For infinitely many particles, the analogue of Lemma 5 is Lemma 6.

Lemma 6 (Kernel Formulation of Infinite-Width) *Equation* (4) *is equivalent to:*

$$\min_{\nu\in\mathcal{P}(\mathbb{R}^d), c\in\mathbb{R}, \alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^n \ell(y_i,(K\alpha)_i+c) + \frac{\lambda}{2}\alpha^\top K\alpha + \frac{\lambda}{2}\int_{\mathbb{R}^d} \|w\|\,\mathrm{d}\nu(w), \tag{9}$$

with $K = \int_{\mathbb{R}^d} K^{(w)}\,\mathrm{d}\nu(w)$ *and and* $K^{(w)} \in \mathbb{R}^{n\times n}$ *is the kernel matrix for kernel* $k^{(B)}$ *and data* $(w^\top x_1,\dots,w^\top x_n)$. *At the optimum, the support of* μ *from Equation* (4) *can be obtained from that of* ν *from Equation* (9) *by normalising all vectors.*

Notice that there is shift in spaces, as ν is a probability distribution on $\mathbb{R}^d$, whereas μ was a probability distribution on $\mathcal{S}^{d-1}$. The proof is provided in Appendix A.2.2.

2.4.1 Kernel Perspective

Lemma 5 shows that we are solving a regularised kernel ridge regression problem where the kernel $\frac{1}{m}\sum_{j=1}^m (|w_j^\top x| + |w_j^\top x'| - |w_j^\top(x-x')|)/2$ is also learnt through the weights $(w_j)_{j\in[m]}$, and the third term $\frac{\lambda}{2}\frac{1}{m}\sum_{j=1}^m \|w_j\|$ serves as a penalty to improve kernel learning.

The homogeneity of the kernel $k^{(B)}$ leads to well-behaved optimisation, as we discuss in Section 3.2 and see in Experiment 1 in Section 5.2. The kernel matrix K is indeed positively 1-homogeneous in the particles $(w_j)_{j\in[m]}$. If we had chosen $\mathcal{H}$ to be the RKHS corresponding to the exponential kernel (or the Gaussian kernel), we would have faced

the challenge of learning the kernel $\sum_{j=1}^{m} e^{-|w_j^\top (x-x')|}$, which exhibits a complex and non-homogeneous dependency on the weights $(w_j)_{j\in[m]}$. By using the Brownian kernel instead of the exponential kernel, we only slightly change the regularisation, regularising with $\int_{\mathbb{R}}(g')^2$ instead of $\int_{\mathbb{R}} g^2 + \int_{\mathbb{R}}(g')^2$ while making the optimisation more tractable.

Compared to multiple kernel learning, BKerNN offers notable advantages. MKL involves combining several predefined kernels, which is prone to overfitting as the number of kernels increases. Additionally, selecting the optimal kernel combination is challenging and often requires sophisticated algorithms. In contrast, BKerNN adapts the kernel through the learned weights $(w_j)_{j\in[m]}$, making the optimisation process simpler and more efficient, as discussed in Section 3.

2.4.2 Neural Network Perspective

Our architecture can also be interpreted as a special type of neural network with one hidden layer. Recall that $\mathcal{F}_\infty$ is inspired by neural networks as it involves linear components w followed by a non-linear part. In neural networks, this non-linear part is typically $\eta\sigma(\cdot)$, which we replaced with $g_w(\cdot) \in \mathcal{H}$ in our setting. The functions in $\mathcal{F}_m$ are expressed similarly with the number of particles m equivalent to the number of neurons in the hidden layer.

As we discuss in Section 3.1, we learn the weights $(w_j)_{j\in[m]}$ through gradient descent, while the functions $(g_j)_{j\in[m]}$ are learned explicitly, leveraging a closed-form solution. This approach resonates with the work of Marion and Berthier (2023) and Bietti et al. (2023). Marion and Berthier (2023) examines a one-hidden layer neural network where the step-sizes for the inner layer are much smaller than those for the outer layer. They prove that the gradient flow converges to the optimum of the non-convex optimisation problem in a simple univariate setting and that the number of neurons does not need to be asymptotically large, which is a stronger result than the usual study of mean-field regimes or neural tangent kernel. Bietti et al. (2023) consider learning the link function in a non-parametric way infinitely faster than the low-rank projection subspace, which resonates with our method, although they focus Gaussian data.

We have also established that the function space $\mathcal{F}_\infty$ with bounded Ω_0 is more extensive than the space of neural networks with ReLU activations in Section 2.3. In Section 3.2, we demonstrate that this enlargement is compatible with efficient optimisation.

2.5 Other Penalties

We now present other penalties designed to achieve different effects. The three terms in Equation (8) correspond to the empirical risk, the standard penalty from KRR on the RKHS norm of the function, and an extra regularisation term on the learnt kernel weights. This additional term, $\frac{\lambda}{2m}\sum_{j=1}^{m}\|w_j\|$, originates from the penalty $\Omega_0(f)$ in Equation (2). However, we can explore other penalties on $w_1, \dots, w_m$ that induce various additional sparsity effects, even if they do not directly correspond to penalties on $f \in \mathcal{F}_m$. Let $W \in \mathbb{R}^{d\times m}$ be the matrix with $(w_1, \dots, w_m)$ as columns, denote by $W^{(a)}$ the a-th row of W, and let $W = USV^\top$ be its singular value decomposition, with S a diagonal matrix composed of $S_1, \dots, S_{\min(m,d)}$. Recall that ν is a probability distribution on $\mathbb{R}^d$.

1. **Basic penalty:** $\Omega_{\text{basic}}(w_1, \dots, w_m) = \frac{1}{2m} \sum_{j=1}^m \|w_j\|$, which we discussed in Section 2.1. In the continuous setting, it corresponds to $\frac{1}{2} \int_{\mathbb{R}^d} \|w\| \mathrm{d}\nu(w)$. This penalty, which does not target any specific pattern in the data-generating mechanism, is the one for which we provide theoretical results in Section 4. However, it does not work as well in practive as the following penalties.

2. **Variable penalty:** $\Omega_{\text{variable}}(w_1, \dots, w_m) = \frac{1}{2} \sum_{a=1}^{\min(m,d)} \big(\frac{1}{m} \sum_{j=1}^m (w_j)_a^2\big)^{1/2}$, which is also equal to $\frac{1}{2\sqrt{m}} \sum_{a=1}^{\min(m,d)} \|W^{(a)}\|_2$. This penalty, inspired by the group Lasso (Yuan and Lin, 2006), is designed for variable selection, pushing quantities $\|W^{(a)}\|_2$ towards zero, thus encouraging dependence on a few variables. In the continuous setting, it corresponds to $\frac{1}{2} \sum_{a=1}^{\min(m,d)} \big(\int_{\mathbb{R}^d} |w_a|^2 \mathrm{d}\nu(w)\big)^{1/2}$.

3. **Feature penalty:** $\Omega_{\text{feature}}(w_1, \dots, w_m) = \frac{1}{2} \operatorname{tr}\big(\big(\frac{1}{m} \sum_{j=1}^m w_j w_j^\top\big)^{1/2}\big)$, which is also equal to $\frac{1}{2} \sum_{a=1}^{\min(m,d)} \frac{S_a}{\sqrt{m}}$ and to the nuclear norm of W divided by $2\sqrt{m}$. It is used for feature learning as it is a convex relaxation of the rank, encouraging W to have low rank and thus dependence on only a few linear transformations of the data. Regularisation using the nuclear norm in the context of feature learning is well-established in the literature, as demonstrated by Argyriou et al. (2008). It corresponds to $\frac{1}{2} \operatorname{tr}\big(\big(\int_{\mathbb{R}^d} w w^\top \mathrm{d}\nu(w)\big)^{1/2}\big)$ in the continuous setting.

4. **Concave variable penalty:** The concave version of the penalty for variable selection, $\Omega_{\text{concave variable}}(w_1, \dots, w_m) = \frac{1}{2s} \sum_{a=1}^{\min(m,d)} \log\big(1 + \frac{s}{\sqrt{m}} \|W^{(a)}\|_2\big)$, with $s \geq 0$. The appeal of the added concavity is discussed below. In the continuous setting, it corresponds to $\frac{1}{2s} \sum_{a=1}^d \log\big(1 + s \int_{\mathbb{R}^d} (w_a)^2 \mathrm{d}\nu(w)\big)^{1/2}\big)$.

5. **Concave feature penalty:** The concave version of the penalty intended for feature learning, $\Omega_{\text{concave feature}}(w_1, \dots, w_m) = \frac{1}{2s} \sum_{a=1}^{\min(m,d)} \log\big(1 + \frac{s}{\sqrt{m}} S_a\big)$ for feature selection, with $s \geq 0$. The appeal of the added concavity is discussed below. In the continuous setting it corresponds to $\frac{1}{2s} \sum_{a=1}^d \log\big(1 + s\big(\big(\int_{\mathbb{R}^d} w w^\top \mathrm{d}\nu(w)\big)^{1/2}\big)_{a,a}\big)$.

The first penalty is convex in both ν and W, making it straightforward to optimise. The second and third penalties, while not convex in ν, are convex in W due to the presence of squared and square root terms on the components of W, easing optimisation in the m particles setting. The fourth and fifth penalties are neither convex in ν nor W, instead, they are concave in W. As s approaches zero, these penalties revert to their non-concave versions. Convex penalties, while easier to handle, can be detrimental by diminishing relevant variables or features to achieve sparsity. Mitigating this effect can involve retraining with the selected variables/features or employing concave penalties, which is the choice we made here. Although concave penalties are more complex to analyse, they often yield better performance because they drive the solution towards the boundary, promoting sparsity (Fan and Li, 2001; Bach et al., 2012). We discuss the impact of the choice of regularisation in Experiment 3 in Section 5.3.

3 Computing the Estimator

In this section, we detail the process of computing the estimator for each of the penalties presented in Section 2.5. We then discuss the importance of the homogeneity of the Brownian kernel and how the optimisation with particles relates to the continuous setting.

3.1 Optimisation Procedure

In this section, we focus on the square loss $\ell(y, y') = \frac{1}{2}(y - y')^2$, which allows for explicit computations. However, the method can be extended to other loss functions using gradient-based techniques, (see Bach, 2024, Chapter 5). Recalling Equation (8) and the penalties described in Section 2.5, the optimisation problem we aim to solve is

$$\min_{w_1,\dots,w_m \in \mathbb{R}^d, c \in \mathbb{R}, \alpha \in \mathbb{R}^n} \frac{1}{2n} \|Y - K\alpha - c\mathbb{1}_n\|_2^2 + \frac{\lambda}{2}\alpha^\top K\alpha + \lambda \Omega_{\text{weights}}(w_1, \dots, w_m), \tag{10}$$

where $K = \frac{1}{m}\sum_{j=1}^m K^{(w_j)}$ and Ω_{weights} represents any of the penalties from Section 2.5.

To solve this problem, we alternate between minimisation with respect to α and c, which is done in closed-form, and minimisation with respect to $w_1, \dots, w_m$ which is done using one step of proximal gradient descent.

3.1.1 Fixed Particles $w_1, \dots, w_m$

When the weights $w_1, \dots, w_m$ are fixed, the kernel matrix K is also fixed, allowing us to find the solution for the constant c and the coefficients α in closed-form. By centring both the kernel matrix and the response Y, we transform the problem into a classical kernel ridge regression problem, for which explicit solutions are well-known.

Lemma 7 (Optimisation for Fixed Particles) *For fixed $w_1, \dots, w_m$ and hence a fixed K, define*

$$G(w_1, \dots, w_m) := \min_{\alpha \in \mathbb{R}^n, c \in \mathbb{R}} \frac{1}{2n} \|Y - K\alpha - c\mathbb{1}_n\|_2^2 + \frac{\lambda}{2}\alpha^\top K\alpha.$$

The optimisation problem defining G is solved by

$$\alpha = (\tilde{K} + n\lambda I)^{-1}\tilde{Y} \quad \text{and} \quad c = \frac{\mathbb{1}^\top Y}{n} - \frac{\mathbb{1}^\top K\alpha}{n},$$

where $\tilde{K} := \Pi K \Pi$ and $\tilde{Y} := Y - \frac{\mathbb{1}\mathbb{1}^\top Y}{n}$, with $\Pi = I - \frac{\mathbb{1}\mathbb{1}^\top}{n}$ being the centring matrix. The objective value is then

$$G(w_1, \dots, w_m) = \frac{\lambda}{2}\tilde{Y}^\top(\tilde{K} + \lambda n I)^{-1}\tilde{Y}.$$

The proof is provided in Appendix A.3.1. Lemma7 allows us to optimise α and c explicitly during the optimisation process. The complexity of this step is $O(n^3 + n^2 d)$, which can be challenging when the sample size n is large, a common drawback of kernel methods. However, techniques like the Nyström method (Drineas and Mahoney, 2005), which approximates the kernel matrix, can help mitigate this issue. Alternatively, we could use gradient descent

techniques, but as shown in Marion and Berthier (2023), it may be beneficial to learn the weights from the hidden layer to the output layer (corresponding to learning $g_1, \dots, g_m$ and hence α) with a much larger step-size than the weights from the input layer to the hidden layer (corresponding to learning $w_1, \dots, w_m$). Learning α and c explicitly represents the limit of this two-timescale regime.

3.1.2 Proximal Step to Optimise the Weights $w_1, \dots, w_m$

Next, we focus on optimising $w_1, \dots, w_m$ while keeping c and α fixed. The goal is to solve

$$\min_{w_1,\dots,w_m \in \mathbb{R}^d} G(w_1, \dots, w_m) + \lambda \Omega_{\text{weights}}(w_1, \dots, w_m), \tag{11}$$

where the dependence on $(w_j)_{j\in[m]}$ in the first term is through the kernel matrix K. Note that G is convex in K but not in $w_1, \dots, w_m$. Additionally, G is differentiable almost everywhere, except where $w_j^\top (x_i - x_{i'})$ for some $j \in [m], i \neq i' \in [n]$. However, standard practice assumes that these non-differentiabilities average out with many data points. Meanwhile, the penalties Ω_{weights} are not differentiable at certain fixed points, independently of the data, similarly to the Lasso penalty. Therefore, we use proximal gradient descent to solve Equation (11). With a step-size $\gamma > 0$, this involves minimising

$$\sum_{j=1}^m \frac{\partial G}{\partial w_j}(w^{\text{old}})^\top (w_j - w_j^{\text{old}}) + \frac{1}{2\gamma} \sum_{j=1}^m \|w_j - w_j^{\text{old}}\|_2^2 + \lambda \Omega_{\text{weights}}(w_1, \dots, w_m),$$

over $w_1, \dots, w_m \in \mathbb{R}^d$. This corresponds to the simultaneous proximal gradient descent steps $w_j \leftarrow \text{prox}_{\lambda\gamma\Omega}(w_j - \gamma \frac{\partial G}{\partial w_j})$. We therefore compute the gradient and the proximal operator. For the gradient, we have the following lemma.

Lemma 8 (Gradient of G) *Let $j \in [m]$, then*

$$\frac{\partial G}{\partial w_j} = \frac{\lambda}{4} \frac{1}{m} \sum_{i,i'=1}^n z_i z_{i'} \operatorname{sign}(w_j^\top (x_i - x_{i'}))(x_i - x_{i'}),$$

where $z = (\tilde{K} + n\lambda I)^{-1} \tilde{Y}$.

The proof is in Appendix A.3.2. Note that G is not differentiable around 0, which is also the case of common activation functions in neural networks such as the ReLU, but this is not an issue in practice.

Next, we compute the proximal operator for the described penalties. Recall the definition of the proximal operator

$$\text{prox}_\Omega(W) = \operatorname*{arg\,min}_{(u_1,\dots,u_m)\in\mathbb{R}^{d\times m}} \frac{1}{2} \sum_{j=1}^m \|w_j - u_j\|_2^2 + \Omega(u_1, \dots, u_m).$$

We use $W \in \mathbb{R}^{d\times m}$ and $(w_1, \dots, w_m)$ interchangeably, with $W = USV^\top$ (SVD). We denote the rows of W by $W^{(a)}$ as before. The following lemma provides the proximal operators.

Lemma 9 (Proximal Operators) *We describe the proximal operators.*

1. *For* $\Omega_{\text{basic}}(W) = \frac{1}{2m}\sum_{j=1}^m \|w_j\|$, *then* $\left(\text{prox}_{\lambda\gamma\Omega}(W)\right)_j = \left(1 - \frac{\lambda\gamma}{2m}\frac{1}{\|w_j\|}\right)_+ w_j$.
2. *For* $\Omega_{\text{variable}}(W) = \frac{1}{2\sqrt{m}}\sum_{a=1}^d \|W^{(a)}\|_2$, $(\text{prox}_{\lambda\gamma\Omega}(W))^{(a)} = \left(1 - \frac{\lambda\gamma}{2\sqrt{m}}\frac{1}{\|W^{(a)}\|_2}\right)_+ W^{(a)}$.
3. *For* $\Omega_{\text{feature}}(W) = \frac{1}{2}\text{trace}\left(\left(\frac{1}{m}\sum_{j=1}^m w_j w_j^\top\right)^{1/2}\right)$, *then we have* $\text{prox}_{\lambda\gamma\Omega}(W) = U\tilde{S}V^\top$ *with* $\tilde{S} = \left(1 - \frac{\lambda\gamma}{2\sqrt{m}|S|}\right)_+ S$.
4. *For* $\Omega_{\text{concave variable}}(W) = \frac{1}{2s}\sum_{a=1}^d \log\left(1 + \frac{s}{\sqrt{m}}\|W^{(a)}\|_2\right)$, *then with* c *obtained from* $(\|W^{(a)}\|_2)_{a\in[d]}$ *by an explicit (albeit lengthy) formula* $(\text{prox}_{\lambda\gamma\Omega}(W))^{(a)} = cW^{(a)}$.
5. *For* $\Omega_{\text{concave feature}}(W) = \frac{1}{2s}\sum_{a=1}^d \log\left(1 + \frac{s}{\sqrt{m}}S_a\right)$, *then with* c *which obtained from* S *by an explicit (albeit lengthy) formula* $\text{prox}_{\lambda\gamma\Omega}(W) = U\tilde{S}V^\top$ *with* $\tilde{S} = cS$.

The proof is in Appendix A.3.3. Each proximal step is easy to compute using the explicit formulas above, with complexities $O(md)$ for the basic, variable, and concave variable cases, and $O(md\min(m,d))$ for the feature and concave feature cases, due to the SVD computation.

3.1.3 Algorithm Pseudocode

We now have all the components necessary to provide the pseudocode (Algorithm 1) of the proposed method BKerNN, specifically for the square loss. For other losses, the main difference is that α and c might not be solvable in closed-form and would need to be computed through alternative methods such as gradient descent.

Data: $X, Y, m, \lambda, \gamma, \Omega_{\text{weights}}$
Result: $w_1, \ldots, w_m, c, \alpha$
$W = (w_1, \ldots, w_m) \in \mathbb{R}^{d\times m} \leftarrow \left(\mathcal{N}(0, 1/d)\right)^{d\times m}$;
for $i \in [n_{\text{iter}}]$ **do**
  Compute K;
  $\alpha \leftarrow (\tilde{K} + n\lambda I)^{-1}\tilde{Y}, c \leftarrow \frac{\mathbb{1}^\top Y}{n} - \frac{\mathbb{1}^\top}{n}K\alpha$;
  Compute $\frac{\partial G}{\partial W}$;
  $\gamma \leftarrow \gamma \times 1.5$;
  while $G(\text{prox}_{\lambda\gamma\Omega}(W - \gamma\frac{\partial G}{\partial W}) > G(W) - \gamma\frac{\partial G}{\partial W}\cdot G_\gamma(W) + \frac{\gamma}{2}\|G_\gamma(W)\|_2^2$ **do**
    $\gamma \leftarrow \gamma/2$;
  end
  $W \leftarrow \text{prox}_{\lambda\gamma\Omega}(W - \gamma\frac{\partial G}{\partial W})$;
end

Algorithm 1: BKerNN pseudocode

To select the step-size γ for the proximal gradient descent step appropriately, we use a backtracking line search, assuming G is locally Lipschitz. Starting with the previous step-size, we multiply it by 1.5. If the backtracking condition is not satisfied, we divide γ by 2 and repeat. The backtracking condition is that $G(\text{prox}_{\lambda\gamma\Omega}(W - \gamma\frac{\partial G}{\partial W}))$ should be smaller

than $G(W) - \gamma \frac{\partial G}{\partial W} \cdot G_\gamma(W) + \frac{\gamma}{2}\|G_\gamma(W)\|_2^2$, where $G_\gamma(W) = \big(W - \mathrm{prox}_{\lambda\gamma\Omega}(W - \gamma\frac{\partial G}{\partial W})\big)/\gamma$. This method was taken from Beck (2017).

With the outputted $w_1, \dots, w_m$, c, and α from the algorithm, the estimator is the function $\hat{f}_\lambda$ defined as $\hat{f}_\lambda(x) = c + \sum_{i=1}^n \alpha_i \sum_{j=1}^m \frac{1}{m}(|w_j^\top x_i| + |w_j^\top x| - |w_j^\top (x - x_i)|)/2$. This formulation enables us to perform predictions on new data points and facilitates the extraction of meaningful linear features through the learned weights $(w_j)_{j\in[m]}$. Remark that we do not take into account the optimisation error in the rest of the paper.

3.2 Convergence Guarantees on Optimisation Procedure

In this section, we discuss the convergence properties of the optimisation procedure. Although we do not provide a formal proof due to differentiability issues, we highlight the importance of the homogeneity of the Brownian kernel and present arguments suggesting the robustness of the optimisation process.

We aim to apply the insights from Chizat and Bach (2022) and Chizat and Bach (2018), which state that under certain assumptions, in the limit of infinitely many particles and an infinitely small step-size, gradient descent optimisation converges to the global optimum of the infinitely-many particles problem. Key assumptions include convexity with respect to the probability distribution in the and homogeneity of a specific quantity Ψ, which we define below. We reformulate our problem in line with Chizat and Bach (2022).

Considering the square loss with the basic penalty Ω_{basic}, the optimisation problem with m particles from Equation (10) can be rewritten as

$$\begin{aligned}
&\min_{w_1,\dots,w_m\in\mathbb{R}^d} \left(\inf_{\alpha\in\mathbb{R}^n, c\in\mathbb{R}} \frac{1}{2n}\|Y - K\alpha - c\mathbb{1}_n\|_2^2 + \frac{\lambda}{2}\alpha^\top K\alpha + \frac{\lambda}{2m}\sum_{j=1}^m \|w_j\| \right) \\
&= \min_{w_1,\dots,w_m\in\mathbb{R}^d} \left(\frac{\lambda}{2}\tilde{Y}^\top \big(\tilde{K} + \lambda n I\big)^{-1}\tilde{Y} + \frac{\lambda}{2m}\sum_{j=1}^m \|w_j\| \right),
\end{aligned}$$

where $K = \frac{1}{m}\sum_{j=1}^m K^{(w_j)}$ is the final kernel matrix, $K^{(w)} \in \mathbb{R}^{n\times n}$ is the kernel matrix for kernel $k^{(B)}$ with projected data $(w^\top x_1, \dots, w^\top x_n)$, $\Pi = I_n - \mathbb{1}_n\mathbb{1}_n^\top$ is the centring matrix, while $\tilde{Y} = \Pi Y$ is the centred output, and $\tilde{K} = \Pi K \Pi$ is the centred kernel matrix. We solve this using proximal gradient descent. For the continuous case, the problem is

$$\begin{aligned}
&\min_{\nu\in\mathcal{P}(\mathbb{R}^d)} \left(\inf_{\alpha\in\mathbb{R}^n, c\in\mathbb{R}} \frac{1}{2n}\|Y - K\alpha - c\mathbb{1}_n\|_2^2 + \frac{\lambda}{2}\alpha^\top K\alpha + \frac{\lambda}{2}\int_{\mathbb{R}^d} \|w\| \,\mathrm{d}\nu(w) \right) \\
&= \min_{\nu\in\mathcal{P}(\mathbb{R}^d)} \left(\frac{\lambda}{2}\tilde{Y}^\top \big(\tilde{K} + \lambda n I\big)^{-1}\tilde{Y} + \frac{\lambda}{2}\int_{\mathbb{R}^d} \|w\| \,\mathrm{d}\nu(w) \right),
\end{aligned}$$

where $K = \int_{\mathbb{R}^d} K^{(w)} \,\mathrm{d}\nu(w)$ and ν is a probability measure on $\mathbb{R}^d$.

In both cases we minimise $F(\nu)$ (defined right below) over $\mathcal{P}(\mathbb{R}^d)$ for the continuous case and over $\mathcal{P}_n(\mathbb{R}^d)$, which is the set of probability distributions anchored at n points on $\mathbb{R}^d$, in the m-particles case. F is defined as

$$F(\nu) := Q\left(\int_{\mathbb{R}^d} \Psi(w) \,\mathrm{d}\nu(w)\right),$$

where $Q : \mathbb{R}^{n\times n} \times \mathbb{R} \to \mathbb{R}$, $Q(K, c') = \frac{\lambda}{2}\tilde{Y}^\top \left(\tilde{K} + \lambda n I\right)^{-1}\tilde{Y} + \frac{\lambda}{2}c'$, and $\Psi : \mathbb{R}^d \to \mathbb{R}^{n\times n} \times \mathbb{R}$, $\Psi(w) = (K^{(w)}, \|w\|)$. Note that Ψ is indeed positively 1-homogeneous, as the necessary condition $\forall w \in \mathbb{R}^d, \forall \kappa > 0,\ \Psi(\kappa w) = \kappa\Psi(w)$ is verified. Moreover, Q is convex in ν, indicating the optimisation is well-posed (while we perform computations for the square loss, we would also obtain a convex function for any convex loss).

Our method employs proximal gradient descent instead of basic gradient descent, which is acceptable as both methods approximate the differential equation arising in the infinitesimal step-size limit. Gradient descent is an explicit method, whereas proximal gradient descent combines implicit and explicit updates (Süli and Mayers, 2003). Moreover, it allows to deal efficiently with the non-smoothness of the sparsity-inducing penalties (no additional cost and improved convergence behaviour).

While our framework aligns with Chizat and Bach (2022), we cannot directly apply their results due to the non-differentiability of Ψ around zero, a common issue in such analyses. Despite this, our setup meets the crucial assumptions of convexity in Q and the homogeneity of Ψ. See Experiment 1 in Section 5.2 for a numerical evaluation of the practical significance of the homogeneity assumption.

4 Statistical Analysis

In this section our objective is to obtain high-probability bounds on the expected risk of the BKerNN estimator to understand its generalisation capabilities. To achieve this, we bound the Gaussian complexity (a similar concept to the Rademacher complexity,) of the sets $\{f \in \mathcal{F}_\infty \mid \max(f(0), \Omega_0(f)) \leq D\}$ for $D > 0$. Recall that $\mathcal{F}_\infty$ is defined in Definition 1. We begin by introducing the Gaussian complexity in Definition 10, followed by Lemma 11, which is used to simplify the quantities for subsequent bounding. We then bound the Gaussian complexities using two distinct techniques in Sections 4.1.1 and 4.1.2. The first technique yields a dimension-dependent bound with better complexity in sample size, while the second provides a dimension-independent bound. Finally, in Section 4.2, we derive the high-probability bound on the expected risk of BKerNN with explicit rates for data with subgaussian square-rooted norm, using an extension of McDiarmid's inequality from Meir and Zhang (2003), before detailing the data-dependent quantities of the rates. All of these results require few assumptions on the problem, and on the data-generating mechanism in particular.

While our method resembles multiple kernel learning, the theoretical results from MKL, which are often related to Rademacher chaos (e.g., Lanckriet et al., 2004; Ying and Campbell, 2010) are not directly applicable. This is because, in our approach, the learned weights are multi-dimensional and embedded within the kernel, rather than being simple scalar weights used to combine predefined kernels. Thus, the unique structure of our model requires different theoretical considerations.

4.1 Gaussian Complexity

Recall that the estimator BKerNN is defined as

$$\hat{f}_\lambda := \operatorname*{arg\,min}_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f(x_i)) + \lambda\Omega(f)$$

where $\mathcal{F}$ is $\mathcal{F}_m := \{f \mid f(x) = c + \frac{1}{m}\sum_{j=1}^m g_j(w_j^\top x), w_j \in \mathcal{S}^{d-1}, g_j \in \mathcal{H}, c \in \mathbb{R}\}$ in practice for optimisation and $\mathcal{F}_\infty := \{f \mid f(x) = c + \int_{\mathcal{S}^{d-1}} g_w(w^\top x)\,\mathrm{d}\mu(w), g_w \in \mathcal{H}, \mu \in \mathcal{P}(\mathcal{S}^{d-1}), c \in \mathbb{R}\}$ for statistical analysis. Although we considered various penalties in Section 2.5, here we focus on $f \in \mathcal{F}_\infty$ with $\Omega(f) = \max(\Omega_0(f), c)$, where $\Omega_0(f)$ was defined as

$$\Omega_0(f) = \inf_{c\in\mathbb{R},\mu\in\mathcal{P}(\mathcal{S}^{d-1}),(g_w)_w\in\mathcal{H}^{\mathcal{S}^{d-1}}} \int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}}\,\mathrm{d}\mu(w),$$

such that $f(\cdot) = c + \int_{\mathcal{S}^{d-1}} g_w(w^\top\cdot)\mathrm{d}\mu(w)$ and corresponds to the basic penalty for Ω_{weights}. This is made possible through a well-defined mean-field limit; we leave the other penalties to future work.

We now introduce the concept of Gaussian complexity (for more details, see Bartlett and Mendelson, 2002).

Definition 10 (Gaussian Complexity) *The Gaussian complexity of a set of functions $\mathcal{G}$ is defined as*

$$G_n(\mathcal{G}) := \mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\sup_{f\in\mathcal{G}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i f(x_i)\right),$$

where ε is a centred Gaussian vector with identity covariance matrix, and $\mathcal{D}_n := (x_1, \dots, x_n)$ is the data set consisting of i.i.d. samples drawn from the distribution of the random variable X. Note that it only contains the covariates, not the response.

We aim to bound $G_n(\{f \in \mathcal{F}_\infty \mid \Omega(f) \le D\})$ for some $D > 0$. The discussion on the Gaussian complexity of the space $\mathcal{F}_\infty$ would yield the same bounds if $\mathcal{F}_m$ were considered instead. However, since we demonstrated in Section 3.2 that optimisation in $\mathcal{F}_m$ and optimisation in $\mathcal{F}_\infty$ are closely related, we focus exclusively on $\mathcal{F}_\infty$ in this section.

First, we note that we can study the Gaussian complexity of a simpler class of functions, as indicated by the following lemma, which allows us to deal with the constant and remove the integral present in the definition of $\mathcal{F}_\infty$.

Lemma 11 (Simplification of Gaussian Complexity) *Let $D > 0$. Then,*

$$G_n(\{f \in \mathcal{F}_\infty \mid \Omega(f) \le D\}) \le D\left(\frac{1}{\sqrt{n}} + G_n\right)$$

with $G_n := G_n(\{f \mid f(\cdot) = g(w^\top\cdot), \|g\|_{\mathcal{H}} \le 1, w \in \mathcal{S}^{d-1}\})$.

The proof can be found in Appendix A.4.1. We now need to bound G_n, which we approach in two different ways. First, in Section 4.1.1, we use covering balls on the sphere $\mathcal{S}^{d-1}$, resulting in a dimension-dependent bound. Then, in Section 4.1.2, we approximate functions in $\mathcal{F}_\infty$ by Lipschitz functions, before using a covering argument, leading to a dimension-independent bound at the cost of worst dependency in the sample size n. With these bounds on G_n, we will derive results on the expected risk of the BKernNN estimator, providing explicit rates depending on the upper bounds of G_n, without exponential dependence on dimension.

4.1.1 Dimension-Dependent Bound

First, we note that the supremum over the functions g with $\|g\|_{\mathcal{H}} \leq 1$ can be obtained in closed-form (see Lemma 18 in Appendix A.4.2). This reduces the problem to considering the expectation of a supremum over the sphere, which we address using a covering of $\mathcal{S}^{d-1}$.

Theorem 12 (Dimension-Dependent Bound) *We have*

$$G_n \leq 8\sqrt{\frac{d}{n}}\sqrt{\log(n+1)}\sqrt{\mathbb{E}_X\|X\|^*},$$

where $\|\cdot\|^*$ is the dual norm of $\|\cdot\|$. Recall that $\|\cdot\|$ defines the sphere $\mathcal{S}^{d-1}$.

The bound on the Gaussian complexity obtained here is dimension-dependent due to the covering of the unit ball in $\mathbb{R}^d$, but it has a favourable dependency on the sample size. For ease of exposition, we have replaced the original factor $\sqrt{\log(1+n/(2d))+1/(2d)}+1$ with $8\sqrt{\log(n+1)}$. Recall that $\|\cdot\|^* = \|\cdot\|_2$ for the ℓ_2 sphere and $\|\cdot\|^* = \|\cdot\|_\infty$ for the ℓ_1 sphere. Note that the dependency on the data distribution is explicit and can be easily bounded in different data-generating mechanisms, as discussed in Lemma 17 at the end of Section 4.

Proof [Theorem 12] First, using Lemma 18, we have

$$G_n = \mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\sup_{w\in\mathcal{S}^{d-1}}\frac{\sqrt{\varepsilon^\top K^{(w)}\varepsilon}}{n}\right),$$

where $K^{(w)}$ is the kernel matrix for the Brownian kernel with data $(w^\top x_1,\dots,w^\top x_n)$.

We bound the supremum inside of the expectation using covering balls. Let $M \in \mathbb{N}^*$ and $\mathcal{W}^M$ be such that $\forall w \in \mathcal{S}^{d-1}, \exists \tilde{w} \in \mathcal{W}^M \subset \mathcal{S}^{d-1}$ such that $\|w-\tilde{w}\| \leq \zeta$, i.e., we have a ζ-covering of the sphere with its own norm in d dimensions. Fix $w \in \mathcal{S}^{d-1}$ and $\tilde{w}$ such that $\|w-\tilde{w}\| \leq \zeta$. We then have

$$\begin{aligned}
|\sqrt{\varepsilon^\top K^{(w)}\varepsilon} - \sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon}| &= \left|\left\|\sum_{i=1}^n \varepsilon_i k_{w^\top x_i}\right\|_{\mathcal{H}} - \left\|\sum_{i=1}^n \varepsilon_i k_{\tilde{w}^\top x_i}\right\|_{\mathcal{H}}\right| \\
&\leq \left\|\sum_{i=1}^n \varepsilon_i (k_{w^\top x_i} - k_{\tilde{w}^\top x_i})\right\|_{\mathcal{H}} \leq \sum_{i=1}^n |\varepsilon_i| \cdot \|k_{w^\top x_i} - k_{\tilde{w}^\top x_i}\|_{\mathcal{H}} \\
&= \sum_{i=1}^n |\varepsilon_i| \sqrt{|w^\top x_i - \tilde{w}^\top x_i|} \leq \sum_{i=1}^n |\varepsilon_i| \sqrt{\|w-\tilde{w}\|\|x_i\|^*} \\
&\leq \sqrt{\|w-\tilde{w}\|}\sum_{i=1}^n |\varepsilon_i|\sqrt{\|x_i\|^*} \leq \zeta^{1/2}\sum_{i=1}^n |\varepsilon_i|\sqrt{\|x_i\|^*}.
\end{aligned}$$

Next, we get

$$\sqrt{\varepsilon^\top K^{(w)}\varepsilon} = \sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon} + \sqrt{\varepsilon^\top K^{(w)}\varepsilon} - \sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon} \leq \sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon} + \zeta^{1/2}\sum_{i=1}^n |\varepsilon_i|\sqrt{\|x_i\|^*}.$$

Taking the supremum and dividing by the sample size n,

$$\sup_{w\in\mathcal{S}^{d-1}} \frac{\sqrt{\varepsilon^\top K^{(w)}\varepsilon}}{n} \leq \sup_{\tilde{w}\in\mathcal{W}^M} \frac{\sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon}}{n} + \zeta^{1/2}\sum_{i=1}^n |\varepsilon_i|\sqrt{\|x_i\|^*}. \tag{12}$$

Considering the expectation over ε of Equation (12), we get

$$\mathbb{E}_\varepsilon\left(\sup_{w\in\mathcal{S}^{d-1}} \frac{\sqrt{\varepsilon^\top K^{(w)}\varepsilon}}{n}\right) \leq \mathbb{E}_\varepsilon\left(\sup_{\tilde{w}\in\mathcal{W}^M} \frac{\sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon}}{n}\right) + \zeta^{1/2}\mathbb{E}_\varepsilon\left(\frac{1}{n}\sum_{i=1}^n |\varepsilon_i|\sqrt{\|x_i\|^*}\right).$$

We now handle $\mathbb{E}_\varepsilon\left(\sup_{\tilde{w}\in\mathcal{W}^M} \frac{\sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon}}{n}\right)$ using standard concentration tools for supremum of infinitely many random variables. Consider $t>0$, then

$$\begin{aligned}
\mathbb{E}_\varepsilon\left(\sup_{\tilde{w}\in\mathcal{W}^M} \sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right) &\leq \sqrt{\mathbb{E}_\varepsilon\left(\sup_{\tilde{w}\in\mathcal{W}^M} \varepsilon^\top K^{(\tilde{w})}\varepsilon\right)} \\
&\leq \sqrt{\frac{1}{t}\log\left(\mathbb{E}_\varepsilon\left(e^{t\sup_{\tilde{w}\in\mathcal{W}^M}\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right)\right)} \\
&= \sqrt{\frac{1}{t}\log\left(\mathbb{E}_\varepsilon\left(\sup_{\tilde{w}\in\mathcal{W}^M} e^{t\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right)\right)} \\
&\leq \sqrt{\frac{1}{t}\log\left(\mathbb{E}_\varepsilon\left(\sum_{\tilde{w}\in\mathcal{W}^M} e^{t\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right)\right)} \\
&= \sqrt{\frac{1}{t}\log\left(\sum_{\tilde{w}\in\mathcal{W}^M} \mathbb{E}_\varepsilon\left(e^{t\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right)\right)}.
\end{aligned}$$

Fix $\tilde{w}\in\mathcal{W}^M$ and consider $\mathbb{E}_\varepsilon\left(e^{t\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right)$. Diagonalising $K^{(\tilde{w})}$ to $U_{\tilde{w}}D_{\tilde{w}}U_{\tilde{w}}^\top$, we have that $U_{\tilde{w}}^\top\varepsilon$ is still a Gaussian vector with identity covariance matrix. When t is small enough, i.e., $\forall i\in[n], 2t(D_{\tilde{w}})_i<1$, or $t<\frac{1}{2\max_i(D_{\tilde{w}})_i}$,

$$\begin{aligned}
\mathbb{E}_\varepsilon\left(e^{t\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right) = \mathbb{E}_\varepsilon\left(e^{t\sum_{i=1}^n (D_{\tilde{w}})_i\varepsilon_i^2}\right) &= \prod_{i=1}^n \mathbb{E}_\varepsilon(e^{t(D_{\tilde{w}})_i\varepsilon_i^2}) \\
&= \prod_{i=1}^n \int_{\mathbb{R}} \frac{1}{\sqrt{2\pi}} e^{(t(D_{\tilde{w}})_i - 1/2)\varepsilon_i^2}\,\mathrm{d}\varepsilon_i \\
&= \prod_{i=1}^n \int_{\mathbb{R}} \frac{1}{\sqrt{2\pi}} e^{(2t(D_{\tilde{w}})_i - 1)\frac{\varepsilon_i^2}{2}} = \prod_{i=1}^n (1-2t(D_{\tilde{w}})_i)^{-1/2}.
\end{aligned}$$

Re-injecting this, we obtain

$$\log\left(\mathbb{E}_\varepsilon\left(e^{t\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right)\right) = \log\left(\prod_{i=1}^n (1-2t(D_{\tilde{w}})_i)^{-1/2}\right) \leq \frac{-1}{2}\sum_{i=1}^n \log(1-2t(D_{\tilde{w}})_i).$$

To bound this further, take $t \leq \frac{1}{4\max((D_{\tilde{w}})_i)}$, which implies both $2t(D_{\tilde{w}})_i < 1/2$ and $-\log(1-2t(D_{\tilde{w}})_i) \leq 4t(D_{\tilde{w}})_i$, leading to

$$\log\left(\mathbb{E}_\varepsilon\left(e^{t\varepsilon^\top K^{(\tilde{w})}\varepsilon}\right)\right) \leq 2t\sum_{i=1}^n (D_{\tilde{w}})_i \leq 2t\,\mathrm{tr}(K^{(\tilde{w})}) \leq 2t\sum_{i=1}^n \|x_i\|^*.$$

Taking $t \leq \min_{\tilde{w}\in\mathcal{W}^M} \frac{1}{4\max((D_{\tilde{w}})_i)}$, we obtain

$$\mathbb{E}_\varepsilon\left(\sup_{\tilde{w}\in\mathcal{W}^M} \frac{\sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon}}{n}\right) \leq \frac{1}{n}\sqrt{\frac{1}{t}\log\left(Me^{2t\sum_{i=1}^n \|x_i\|^*}\right)} \leq \frac{1}{n}\sqrt{\frac{1}{t}\left(\log M + 2t\sum_{i=1}^n \|x_i\|^*\right)}.$$

Taking $t = \frac{1}{4\sum_{i=1}^n \|x_i\|^*}$, which fulfils the previously required conditions, we get

$$\mathbb{E}_\varepsilon\left(\sup_{\tilde{w}\in\mathcal{W}^M} \frac{\sqrt{\varepsilon^\top K^{(\tilde{w})}\varepsilon}}{n}\right) \leq \frac{1}{\sqrt{n}}\sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}\sqrt{4\log M + 2}.$$

In the end, we obtain

$$\mathbb{E}_\varepsilon\left(\sup_{w\in\mathcal{S}^{d-1}} \frac{\sqrt{\varepsilon^\top K^{(w)}\varepsilon}}{n}\right) \leq \frac{1}{\sqrt{n}}\sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}\sqrt{4\log M + 2} + \zeta^{1/2}\sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}},$$

where we have used $\mathbb{E}_\varepsilon|\varepsilon_i| \leq \sqrt{\mathbb{E}_\varepsilon(\varepsilon_i)^2} = 1$ and $\frac{\sum_{i=1}^n \sqrt{\|x_i\|^*}}{n} \leq \sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}$.

We know that $M \leq (1+2/\zeta)^d$ (Wainwright, 2019, Lemma 5.7), yielding

$$\begin{aligned}
\mathbb{E}_\varepsilon\left(\sup_{w\in\mathcal{S}^{d-1}} \frac{\sqrt{\varepsilon^\top K^{(w)}\varepsilon}}{n}\right) &\leq \sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}\left(\frac{\sqrt{4d\log(1+\frac{2}{\zeta})+2}}{\sqrt{n}} + \zeta^{1/2}\right) \\
&\leq \sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}\left(\frac{\sqrt{4d\log(1+\frac{n}{2d})+2}}{\sqrt{n}} + \frac{\sqrt{4d}}{n}\right) \\
&\leq 2\sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}\frac{\sqrt{d}}{\sqrt{n}}\left(\sqrt{\log\left(1+\frac{n}{2d}\right)+\frac{1}{2d}}+1\right) \\
&\leq 4\sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}\frac{\sqrt{d}}{\sqrt{n}}\left(\sqrt{\log\left(1+\frac{n}{2d}\right)}+1\right) \\
&\leq 8\sqrt{\frac{\sum_{i=1}^n \|x_i\|^*}{n}}\frac{\sqrt{d}}{\sqrt{n}}\sqrt{\log(n+1)},
\end{aligned}$$

where to get the second line, we took $\zeta = 4d/n$. By taking the expectation over the data set $\mathcal{D}_n$, since $\mathbb{E}_{\mathcal{D}_n}\left(\sqrt{n^{-1}\sum_{i=1}^n \|x_i\|^*}\right) \leq \sqrt{\mathbb{E}(\|X\|^*)}$, we have the desired result. ■

4.1.2 Dimension-Independent Bound

We now bound the Gaussian complexity with a quantity that does not explicitly depend on the dimension of the data. Recall that we aim to bound

$$G_n = \mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\sup_{\|g\|_{\mathcal{H}}\leq 1, w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i g(w^\top x_i)\right),$$

where ε is a centred Gaussian vector with an identity covariance matrix. First, recall that the functions in $\mathcal{H}$ with norm bounded by 1 are not Lipschitz functions but are instead 1/2-Hölder functions (Lemma 3). Specifically, let $g \in \mathcal{H}, \|g\|_{\mathcal{H}} \leq 1$, then for any $a, b \in \mathbb{R}$, we have $|g(a) - g(b)| \leq \|k_a - k_b\|_{\mathcal{H}} = \sqrt{|a-b|}$.

An interesting result for a fixed 1-Lipschitz function h is that we can apply the contraction principle (Bach, 2024, Proposition 4.3) to the Rademacher complexity. Informally, this yields

$$\mathbb{E}_\varepsilon\left(\sup_{w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i h(w^\top x_i)\right) \leq \mathbb{E}_\varepsilon\left(\sup_{w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i w^\top x_i\right),$$

where exceptionally ε is composed of independent Rademacher variables. The supremum in the second term can then be taken explicitly. We will make use of this idea by first approximating the functions in the unit ball of $\mathcal{H}$ with Lipschitz functions, before using Slepian's lemma (Ledoux and Talagrand, 1991, Corollary 3.14) to obtain similar results on the Gaussian complexity.

Lemma 13 (Lipschitz Approximation) *Let $g \in \mathcal{H}$ with $\|g\|_{\mathcal{H}} \leq 1$, and let $\zeta > 0$. There exists a $(1/\zeta)$-Lipschitz function $g_\zeta : \mathbb{R} \to \mathbb{R}$ with $g_\zeta(0) = 0$ such that $\|g - g_\zeta\|_\infty \leq \zeta$.*

The proof can be found in Appendix A.4.3. This lemma indicates that we can approximate functions in the unit ball of the RKHS $\mathcal{H}$ up to any precision in the infinite norm by Lipschitz functions with a Lipschitz constant equal to the inverse of the precision.

Theorem 14 (Dimension-Independent Bound) *If $\mathcal{S}^{d-1}$ is the ℓ_1 or the ℓ_2 sphere, then*

$$G_n \leq \frac{3}{n^{1/6}}\left((\log 2d)^{1/4}\mathbb{1}_{*=\infty} + \mathbb{1}_{*=2}\right)\left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}\left(\|X_i\|^*\right)^2\right)\right)^{1/4}.$$

Recall that in the ℓ_1 sphere case, $\|\cdot\|^* = \|\cdot\|_\infty$, and in the ℓ_2 case $\|\cdot\|^* = \|\cdot\|_2$. Here, we obtain a bound on the Gaussian complexity that depends only mildly on the data dimension d, either not at all in the case of the ℓ_2 sphere or logarithmically for the ℓ_1 sphere. This means that the estimator BKernNN can be effectively used in high-dimensional settings, where the data dimension may be exponentially large relative to the sample size. This improved dependency on the dimension d comes at the cost of a worse dependency on the sample size n compared to Theorem 12. Note also that there can be an implicit dependency on

the dimension through the data distribution, which we discuss in Lemma 17 at the end of Section 4 under different data-generating mechanisms.

Proof [Theorem 14] By applying Lemma 13, we have for any $\zeta_1 > 0$

$$\begin{aligned}\hat{G}_n &:= \mathbb{E}_\varepsilon \left(\sup_{w\in\mathcal{S}^{d-1}, \|g\|_{\mathcal{H}}\leq 1} \frac{1}{n}\sum_{i=1}^n \varepsilon_i g(w^\top x_i) \right) \\ &= \mathbb{E}_\varepsilon \left(\sup_{w\in\mathcal{S}^{d-1}, \|g\|_{\mathcal{H}}\leq 1} \frac{1}{n}\sum_{i=1}^n \varepsilon_i \left(g_{\zeta_1}(w^\top x_i) + g(w^\top x_i) - g_{\zeta_1}(w^\top x_i) \right) \right) \\ &\leq \mathbb{E}_\varepsilon \left(\sup_{w\in\mathcal{S}^{d-1}, \|g\|_{\mathcal{H}}\leq 1} \frac{1}{n}\sum_{i=1}^n \varepsilon_i g_{\zeta_1}(w^\top x_i) \right) + \|g - g_{\zeta_1}\|_\infty.\end{aligned}$$

We can then change the supremum over the unit ball of $\mathcal{H}$ to a supremum over Lipschitz functions

$$\begin{aligned}\hat{G}_n &\leq \mathbb{E}_\varepsilon \left(\sup_{w\in\mathcal{S}^{d-1},\ g_{\zeta_1}(1/\zeta_1)-\text{Lip},\ g_{\zeta_1}(0)=0} \frac{1}{n}\sum_{i=1}^n \varepsilon_i g_{\zeta_1}(w^\top x_i) \right) + \zeta_1 \\ &= \frac{1}{\zeta_1}\mathbb{E}_\varepsilon \left(\sup_{h\ 1-\text{Lip},\ h(0)=0} \sup_{w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i h(w^\top x_i) \right) + \zeta_1 \\ &= \sqrt{\mathbb{E}_\varepsilon \left(\sup_{h\ 1-\text{Lip},\ h(0)=0} \sup_{w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i h(w^\top x_i) \right)},\end{aligned}$$

by choosing the best ζ_1. Technically, we can restrict ourselves to the following class of function: $\mathcal{F}_{1-\text{Lip}} := \{h : [-\max_{i\in[n]} \|x_i\|^*, \max_{i\in[n]} \|x_i\|^*] \to \mathbb{R} \mid h(0) = 0, h \text{ is } 1-\text{Lipschitz}\}$.

We then use a covering argument. To cover $\mathcal{F}_{1-\text{Lip}}$ up to precision $\zeta_2 > 0$ in $\|\cdot\|_\infty$ norm with M functions from $\mathcal{F}_{1-\text{Lip}}$, one needs $M \leq \left(\frac{8\max_{i\in[n]}\|x_i\|^*}{\zeta_2} + 1\right) 2^{\frac{4\max_{i\in[n]}\|x_i\|^*}{\zeta_2}}$ (Luxburg and Bousquet, 2004, Theorem 17). Let $h_1, \dots h_M$ be such a covering. This yields that

$$\begin{aligned}\hat{G}_n &\leq \sqrt{\mathbb{E}_\varepsilon \left(\sup_{h\in\mathcal{F}_{1-\text{Lip}}} \sup_{w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i h(w^\top x_i) \right)} \\ &\leq \sqrt{\mathbb{E}_\varepsilon \left(\sup_{h\in\{h_1,\dots,h_M\}} \sup_{w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i} \varepsilon_i h(w^\top x_i) \right) + \zeta_2},\end{aligned}$$

by proceeding as with the covering of the unit ball of $\mathcal{H}$. We then use Slepian's lemma (Ledoux and Talagrand, 1991, Corollary 3.14). For $h \in \{h_1, \dots h_M\}, w \in \mathcal{S}^{d-1}$, let

$$X_{h,w} := \frac{1}{n}\sum_{i=1}^n \varepsilon_i h(w^\top x_i) \text{ and } Y_{h,w} = \frac{\sqrt{2}}{n}\sum_{i=1}^n \varepsilon_i w^\top x_i + \sum_{j=1}^M \mathbb{1}_{h=h_j}\tilde{\varepsilon}_j \sqrt{8\frac{\sum_{i=1}^n (\|x_i\|^*)^2}{n^2}},$$

where $\tilde{\varepsilon}$ is a centred Gaussian vector with identity covariance matrix independent of ε. Notice that for $h, \tilde{h} \in \{h_1, \dots h_M\}, w, \tilde{w} \in \mathcal{S}^{d-1}$, we have

$$\begin{aligned}
\mathbb{E}_\varepsilon((X_{h,w} - X_{\tilde{h},\tilde{w}})^2) &= \frac{1}{n^2}\sum_{i=1}^n (h(w^\top x_i) - \tilde{h}(\tilde{w}^\top x_i))^2 \\
&\leq \frac{1}{n^2}\sum_{i=1}^n (h(w^\top x_i) - h(\tilde{w}^\top x_i) + h(\tilde{w}^\top x_i) - \tilde{h}(\tilde{w}^\top x_i))^2 \\
&\leq \frac{2}{n^2}\sum_{i=1}^n (h(w^\top x_i) - h(\tilde{w}^\top x_i))^2 + (h(\tilde{w}^\top x_i) - \tilde{h}(\tilde{w}^\top x_i))^2.
\end{aligned}$$

We can then deal with the two terms separately. For the left term, the fact that h is 1-Lipschitz yields that

$$\frac{2}{n^2}\sum_{i=1}^n (h(w^\top x_i) - h(\tilde{w}^\top x_i))^2 \leq \frac{2}{n^2}\sum_{i=1}^n (w^\top x_i - \tilde{w}^\top x_i)^2.$$

Then, using the fact that $h - \tilde{h}$ is 2-Lipschitz and $h(0) = \tilde{h}(0) = 0$, we have

$$\begin{aligned}
\frac{2}{n^2}\sum_{i=1}^n (h(\tilde{w}^\top x_i) - \tilde{h}(\tilde{w}^\top x_i))^2 &= \frac{2}{n^2}\sum_{i=1}^n (h(\tilde{w}^\top x_i) - \tilde{h}(\tilde{w}^\top x_i) - (h(0) - \tilde{h}(0))^2 \\
&\leq \frac{2}{n^2}\sum_{i=1}^n \mathbb{1}_{h\neq h'} 4(w^\top x_i)^2 \leq \mathbb{1}_{h\neq\tilde{h}} \frac{8}{n^2}\sum_{i=1}^n (\|x_i\|^*)^2.
\end{aligned}$$

All in all $\mathbb{E}_\varepsilon((X_{h,w} - X_{\tilde{h},\tilde{w}})^2) \leq \mathbb{E}_\varepsilon((Y_{h,w} - Y_{\tilde{h},\tilde{w}})^2)$ therefore we can apply Slepian's lemma and obtain

$$\begin{aligned}
&\mathbb{E}_\varepsilon\left(\sup_{h\in\{h_1,\dots,h_M\}, w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i h(w^\top x_i)\right) \\
&\leq \mathbb{E}_\varepsilon\left(\sup_{h\in\{h_1,\dots,h_M\}, w\in\mathcal{S}^{d-1}} \frac{\sqrt{2}}{n}\sum_{i=1}^n \varepsilon_i w^\top x_i + \sum_{j=1}^m \tilde{\varepsilon}_j \mathbb{1}_{h=h_j}\sqrt{8\frac{\sum_{i=1}^n (\|x_i\|^*)^2}{n^2}}\right).
\end{aligned}$$

We then remark that the first term of the expectation does not depend on h and that we can take the supremum over the sphere explicitly, while the second term does not depend on w and we can also take the supremum over $\{h_1, \dots, h_M\}$ explicitly

$$\begin{aligned}
&\mathbb{E}_\varepsilon\left(\sup_{h\in\{h_1,\dots,h_M\}, w\in\mathcal{S}^{d-1}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i h(w^\top x_i)\right) \\
&\leq \mathbb{E}_\varepsilon\left(\sup_{w\in\mathcal{S}^{d-1}} \frac{\sqrt{2}}{n}\sum_{i=1}^n \varepsilon_i w^\top x_i + \sup_{h\in\{h_1,\dots,h_M\}} \sum_{j=1}^m \tilde{\varepsilon}_j \mathbb{1}_{h=h_j}\sqrt{8\frac{\sum_{i=1}^n (\|x_i\|^*)^2}{n^2}}\right) \\
&\leq \mathbb{E}_\varepsilon\left(\sup_{w\in\mathcal{S}^{d-1}} \frac{\sqrt{2}}{n}\sum_{i=1}^n \varepsilon_i w^\top x_i + \sup_{j\in[M]} \tilde{\varepsilon}_j \sqrt{8\frac{\sum_{i=1}^n (\|x_i\|^*)^2}{n^2}}\right) \\
&\leq \mathbb{E}_\varepsilon\left(\left\|\frac{\sqrt{2}}{n}\sum_{i=1}^n \varepsilon_i x_i\right\|^* + \sqrt{8\frac{\sum_{i=1}^n (\|x_i\|^*)^2}{n^2}}\sqrt{2\log M}\right). \qquad (13)
\end{aligned}$$

We consider each term of Equation (13) separately, while also taking expectation with regards to the data set. For the second term, using the bound on M (Luxburg and Bousquet, 2004, Theorem 17) and basic inequalities to simplify the term, we have

$$\begin{aligned}
&\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\sqrt{8\frac{\sum_{i=1}^n(\|x_i\|^*)^2}{n^2}}\sqrt{2\log M}\right)\\
&\quad\leq \mathbb{E}_{\mathcal{D}_n}\left(\sqrt{8\frac{\sum_{i=1}^n(\|x_i\|^*)^2}{n^2}}\sqrt{\frac{4\max_{i\in[n]}\|x_i\|^*}{\zeta_2}\log 2+\log\left(\frac{8\max_{i\in[n]}\|x_i\|^*}{\zeta_2}+1\right)}\right)\\
&\quad\leq \mathbb{E}_{\mathcal{D}_n}\left(8\sqrt{\frac{\sum_{i=1}^n(\|x_i\|^*)^2}{n^2}}\sqrt{\frac{\max_{i\in[n]}\|x_i\|^*}{\zeta_2}}\right)\\
&\quad\leq \frac{8}{\sqrt{n}}\frac{1}{\sqrt{\zeta_2}}\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^{3/2}\right)\leq \frac{8}{\sqrt{n}}\frac{1}{\sqrt{\zeta_2}}\left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)\right)^{3/4}.
\end{aligned}$$

We can reinject, yielding

$$\begin{aligned}
G_n^2&\leq \mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\left\|\frac{\sqrt{2}}{n}\sum_{i=1}^n\varepsilon_i x_i\right\|^*\right)+\frac{8}{\sqrt{n}}\frac{1}{\sqrt{\zeta_2}}\left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)\right)^{3/4}+\zeta_2\\
&\leq \mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\left\|\frac{\sqrt{2}}{n}\sum_{i=1}^n\varepsilon_i x_i\right\|^*\right)+2\left(\frac{8}{\sqrt{n}}\left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)\right)^{3/4}\right)^{2/3}\\
&\leq \mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\left\|\frac{\sqrt{2}}{n}\sum_{i=1}^n\varepsilon_i x_i\right\|^*\right)+2\frac{4}{n^{1/3}}\sqrt{\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)},
\end{aligned}$$

by taking $\zeta_2^{3/2}=\frac{8}{\sqrt{n}}\left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)\right)^{3/4}$ in the second line.

Now for the first term from Equation (13) which we have to deal with still, consider first the case $\|\cdot\|^*=\|\cdot\|_2$ then,

$$\begin{aligned}
\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\|\frac{\sqrt{2}}{n}\sum_{i=1}^n\varepsilon_i x_i\|_2\right)&\leq\sqrt{\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\|\frac{\sqrt{2}}{n}\sum_{i=1}^n\varepsilon_i x_i\|_2^2\right)}\\
&=\frac{\sqrt{2}}{n}\sqrt{\mathbb{E}_{\mathcal{D}_n}\left(\sum_{i=1}^n\|x_i\|_2^2\right)}=\frac{\sqrt{2}}{\sqrt{n}}\sqrt{\mathbb{E}_X\left(\|X\|_2^2\right)}.
\end{aligned}$$

In the other case where $\|\cdot\|^*=\|\cdot\|_\infty$, we can use Boucheron et al. (2013, Theorem 2.5), as for a fixed data set $\mathcal{D}_n$, $\sum_{i=1}^n\varepsilon_i s(x_i)_a$ is a centred Gaussian vector with variance equal to

$\sum_{i=1}^n ((x_i)_a)^2$ which is smaller than $\max_{a\in[d]} \sum_{i=1}^n ((x_i)_a)^2$. This yields that

$$\begin{aligned}
\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\|\frac{\sqrt{2}}{n}\sum_{i=1}^n \varepsilon_i x_i\|_\infty\right) &= \frac{\sqrt{2}}{n}\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\max_{a\in[d]}|\sum_{i=1}^n \varepsilon_i (x_i)_a|\right) \\
&= \frac{\sqrt{2}}{n}\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\max_{a\in[d],s\in\{-1,1\}}\sum_{i=1}^n \varepsilon_i s(x_i)_a\right) \\
&\leq \frac{\sqrt{2}}{n}\mathbb{E}_{\mathcal{D}_n}\left(\max_{a\in[d]}\sqrt{2\sum_{i=1}^n((x_i)_a)^2\log(2d)}\right).
\end{aligned}$$

We then have

$$\begin{aligned}
\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\|\frac{\sqrt{2}}{n}\sum_{i=1}^n \varepsilon_i x_i\|_\infty\right) &\leq \frac{2}{n}\sqrt{\log 2d}\mathbb{E}_{\mathcal{D}_n}\left(\sqrt{\max_{a\in[d]}\sum_{i=1}^n((x_i)_a)^2}\right) \\
&\leq \frac{2}{n}\sqrt{\log 2d}\mathbb{E}_{\mathcal{D}_n}\left(\sqrt{\sum_{i=1}^n\max_{a\in[d]}((x_i)_a)^2}\right) \\
&\leq \frac{2}{n}\sqrt{\log 2d}\mathbb{E}_{\mathcal{D}_n}\left(\sqrt{\sum_{i=1}^n\|x_i\|_\infty^2}\right) \\
&\leq \frac{2}{\sqrt{n}}\sqrt{\log 2d}\sqrt{\mathbb{E}_X\left(\|X\|_\infty^2\right)}.
\end{aligned}$$

This yields that the last term of Equation (13) can be bounded

$$\begin{aligned}
G_n^2 &\leq \left(\frac{2}{\sqrt{n}}\sqrt{\log 2d}\mathbb{1}_{*=\infty} + \frac{\sqrt{2}}{\sqrt{n}}\mathbb{1}_{*=2}\right)\sqrt{\mathbb{E}_X((\|X\|^*)^2)} + 2\frac{4}{n^{1/3}}\sqrt{\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)} \\
&\leq \left(\sqrt{\log 2d}\mathbb{1}_{*=\infty} + \mathbb{1}_{*=2}\right)\frac{8}{n^{1/3}}\sqrt{\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)},
\end{aligned}$$

hence

$$G_n \leq \left(\sqrt{\log 2d}\mathbb{1}_{*=\infty} + \mathbb{1}_{*=2}\right)^{1/4}\frac{3}{n^{1/6}}\left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}(\|x_i\|^*)^2\right)\right)^{1/4},$$

which concludes the proof. ■

4.2 Bound on Expected Risk of Regularised Estimator

We now use the bounds on the Gaussian complexity we have obtained in Section 4.1 to derive a bound on the expected risk of BKerNN. We show that, with explicit rates, the expected risk of our estimator converges with high-probability to that of the minimiser for data with subgaussian norms, which includes both bounded data and data with subgaussian components.

First, we provide a definition of subgaussian real variables, as given by Vershynin (2018).

Definition 15 (Subgaussian Variables) *Let Z be a real-valued (not necessarily centred) random variable. Z is subgaussian with variance proxy σ^2 if and only if*

$$\forall t > 0, \max\left(\mathbb{P}(Z \geq t), \mathbb{P}(Z \leq -t)\right) \leq e^{-\frac{t^2}{2\sigma^2}}.$$

We now present the main theoretical result of the paper.

Theorem 16 (Bound on Expected Risk with High-Probability) *Let the estimator function be $\hat{f}_\lambda := \arg\min_{f \in \mathcal{F}_\infty} \widehat{\mathcal{R}}(f) + \lambda\Omega(f)$. Assume the following:*

1. ***Well-specified model**: The minimiser $f^* := \arg\min_{f \in \mathcal{F}_\infty, \Omega(f) < +\infty} \mathcal{R}(f)$ exists.*
2. ***Convexity of the loss**: For any $(x, y) \in \mathcal{X} \times \mathcal{Y}$. $f \in \mathcal{F}_\infty \to \ell(y, f(x))$ is convex.*
3. ***Lipschitz Condition**: The loss ℓ is L-Lipschitz in its second (bounded) argument, i.e., $\forall y \in \mathcal{Y}, a \in \{f(x) \mid x \in \mathcal{X},\ f \in \mathcal{F}_\infty,\ \Omega(f) \leq 2\Omega(f^*)\}, a \to \ell(y, a)$ is L-Lipschitz.*
4. ***Data Distribution**: The data set $(x_i, y_i)_{i \in [n]}$ consists of i.i.d. samples of the random variable (X, Y) where $1 + \sqrt{\|X\|^*}$ is subgaussian with variance proxy σ^2.*

Then, for any $\delta \in (0, 1)$, with probability larger than $1 - \delta$, for $\lambda = 12L\left(\frac{1}{\sqrt{n}} + G_n\right) + \frac{288L\sigma}{\sqrt{n}}\sqrt{\log\frac{1}{\delta}}$,

$$\mathcal{R}(\hat{f}_\lambda) \leq \mathcal{R}(f^*) + 24\Omega(f^*)L\left(\frac{1}{\sqrt{n}} + G_n + \frac{24\sigma}{\sqrt{n}}\sqrt{\log\frac{1}{\delta}}\right).$$

With the bounds on G_n from Theorem 12 and Theorem 14, recall that if $\|\cdot\|$ is either $\|\cdot\|_2$ or $\|\cdot\|_1$, we have

$$G_n \leq \min\left(\frac{3}{n^{1/6}}\left((\log 2d)^{1/4}\mathbb{1}_{*=\infty} + \mathbb{1}_{*=2}\right)\left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i \in [n]}(\|X_i\|^*)^2\right)\right)^{1/4}, 8\sqrt{\frac{d}{n}}\sqrt{\log(n+1)}\sqrt{\mathbb{E}_X\|X\|^*}\right).$$

Proof [Theorem 16] This proof is primarily based on Bach (2024, Proposition 4.7).

Let f^*_λ be a minimiser of $\mathcal{R}_\lambda := \mathcal{R} + \lambda\Omega$ over $\mathcal{F}_\infty$. Consider the set $\mathcal{C}_\tau := \{f \in \mathcal{F}_\infty \mid \mathcal{R}_\lambda(f) - \mathcal{R}_\lambda(f^*_\lambda) \leq \tau\}$ for some $\tau > 0$ that will be chosen later. $\mathcal{C}_\tau$ is a convex set by the convexity assumption on the loss ℓ.

First, we show that $\mathcal{C}_\tau$ is included in the set $\mathcal{B}_\tau := \{f \in \mathcal{F}_\infty \mid \Omega(f) \leq \Omega(f^*) + \tau/\lambda\}$. This inclusion follows from the optimality of f^* and f^*_λ. Let $f \in \mathcal{C}_\tau$, then

$$\mathcal{R}_\lambda(f) \leq \mathcal{R}_\lambda(f^*_\lambda) + \tau \leq \mathcal{R}_\lambda(f^*) + \tau \leq \mathcal{R}(f) + \lambda\Omega(f^*) + \tau,$$

yielding $f \in \mathcal{B}_\tau$.

Next, set $\tau = \lambda\Omega(f^*)$ with λ to be chosen later. We show that $\hat{f}_\lambda$ belongs to $\mathcal{C}_\tau$ with high probability. If $\hat{f}_\lambda \notin \mathcal{C}_\tau$, since $f^*_\lambda \in \mathcal{C}_\tau$ and $\mathcal{C}_\tau$ is convex, there exists a $\tilde{f}$ in the segment

$[\hat{f}_\lambda, f_\lambda^*]$ and which is on the boundary of $\mathcal{C}_\tau$, i.e. such that $\mathcal{R}_\lambda(\tilde{f}) = \mathcal{R}_\lambda(f_\lambda^*) + \tau$. Since the empirical risk is convex, we have $\widehat{\mathcal{R}}_\lambda(\tilde{f}) \leq \max(\widehat{\mathcal{R}}_\lambda(\hat{f}_\lambda), \widehat{\mathcal{R}}_\lambda(f_\lambda^*)) = \widehat{\mathcal{R}}_\lambda(f_\lambda^*)$. Then,

$$\begin{aligned}\widehat{\mathcal{R}}(f_\lambda^*) - \widehat{\mathcal{R}}(\tilde{f}) - \mathcal{R}(f_\lambda^*) + \mathcal{R}(\tilde{f}) &= \widehat{\mathcal{R}}_\lambda(f_\lambda^*) - \widehat{\mathcal{R}}_\lambda(\tilde{f}) - \mathcal{R}_\lambda(f_\lambda^*) + \mathcal{R}_\lambda(\tilde{f}) \\ &\geq -\mathcal{R}_\lambda(f_\lambda^*) + \mathcal{R}_\lambda(\tilde{f}) = \tau. \end{aligned} \tag{14}$$

Note that $\Omega(\tilde{f}) \leq 2\Omega(f^*)$ and $\Omega(f_\lambda^*) \leq 2\Omega(f^*)$. Combining Lemma 19 and Lemma 21, for $\delta \in (0,1)$, with probability greater than $1-\delta$, we have for all $f \in \mathcal{F}_\infty$ such that $\Omega(f) \leq 2\Omega(f^*)$:

$$\begin{aligned}&\widehat{\mathcal{R}}(f_\lambda^*) - \widehat{\mathcal{R}}(f) - \mathcal{R}(f_\lambda^*) + \mathcal{R}(f) \\ &\quad\leq \mathbb{E}_{\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty, \Omega(f)\leq 2\Omega(f^*)} \widehat{\mathcal{R}}(f) - \mathcal{R}(f) + \sup_{f\in\mathcal{F}_\infty, \Omega(f)\leq 2\Omega(f^*)} \mathcal{R}(f) - \widehat{\mathcal{R}}(f)\right) \\ &\qquad + \Omega(f^*)\frac{96\sqrt{2e}L\sigma}{\sqrt{n}}\sqrt{\log\frac{1}{\delta}} \\ &\quad\leq 12\Omega(f^*)L\left(\frac{1}{\sqrt{n}} + G_n\right) + \Omega(f^*)\frac{96\sqrt{2e}L\sigma}{\sqrt{n}}\sqrt{\log\frac{1}{\delta}}.\end{aligned}$$

Now, choose λ such that $\tau = \lambda\Omega(f^*) \geq 12\Omega(f^*)L\left(\frac{1}{\sqrt{n}} + G_n\right) + \Omega(f^*)\frac{96\sqrt{2e}L\sigma}{\sqrt{n}}\sqrt{\log\frac{1}{\delta}}$. This yields a contradiction with Equation (14). Thus, with such a λ, with probability greater than $1-\delta$, we have $\hat{f}_\lambda \in \mathcal{C}_\tau$, hence

$$\begin{aligned}\mathcal{R}_\lambda(\hat{f}_\lambda) &\leq \mathcal{R}_\lambda(f_\lambda^*) + \lambda\Omega(f^*), \\ \mathcal{R}(\hat{f}_\lambda) &\leq \mathcal{R}(f^*) + 2\lambda\Omega(f^*).\end{aligned}$$

For $\lambda = 12L\left(\frac{1}{\sqrt{n}} + G_n\right) + \frac{288L\sigma}{\sqrt{n}}\sqrt{\log\frac{1}{\delta}}$, this yields

$$\mathcal{R}(\hat{f}_\lambda) \leq \mathcal{R}(f^*) + \Omega(f^*)\left(24L\left(\frac{1}{\sqrt{n}} + G_n\right) + \frac{576L\sigma}{\sqrt{n}}\sqrt{\log\frac{1}{\delta}}\right).$$

■

We now provide insightful comments on Theorem 16. We first remark that the result could be proven more directly for bounded data using McDiarmid's inequality, resulting in a better constant.

The chosen λ does not depend on unknown quantities such as $\Omega(f^*)$, but only on known quantities such as the Lipschitz constant of the loss, the sample size, or the dimension of the data. This allows λ to be explicitly chosen for a fixed probability δ, although it is usually computed through cross-validation.

Classical losses typically satisfy our assumptions. For instance, the square loss is always convex and L-Lipschitz if the data and response are bounded, with $L = 2\sup_{y\in\mathcal{Y}}|y| + 4\Omega(f^*)\sup_{x\in\mathcal{X}}\|x\|^*$. Similarly, the logistic loss is always convex and L-Lipschitz with $L = 1$ (in the context of outputs in $\{-1, 1\}$).

Our approach stands out by requiring minimal assumptions on the data-generating mechanism, which is less restrictive compared to other methodologies in the multi-index model domain. This emphasis on general applicability is also why we do not include feature recovery results, as such outcomes typically necessitate strong assumptions about the data and often require prior knowledge of the distribution.

The rates obtained depend explicitly on the dimension of the data through the bound on the Gaussian complexity. Considering the first term in the minimum, we observe that the bound is independent (up to logarithmic factors) of the data dimension, making BKerNN suitable for high-dimensional problems. However, this bound has a less favourable dependency on the sample size compared to the dimension-dependent bound, which is the second term in the minimum. We conjecture that the actual rate has the best of both worlds, achieving an explicit dependency on dimension d and sample size n of $n^{-1/2}$ (up to logarithmic factors).

Comparing the rate between BKerNN, neural networks with ReLU activations, and kernel methods, we find that in well-specified settings (where the Bayes estimator belongs to each function space considered), KRR yields a $O(n^{-1/2})$ rate independent of dimension, but require very smooth functions, for example, a Sobolev space of order s (i.e. the derivatives up to order s are square integrable) is only a RKHS if $s > d/2$ (Bach, 2024, Chapter 7). Neural networks with ReLU activation achieve a similar rate with fewer constraints, as their function space is typically larger than RKHS spaces (Bach, 2024, Chapter 9).

If the model is not well-specified but we consider the Bayes predictor f^* to be Lipschitz continuous, the rates for neural networks with ReLU activation and bounded corresponding Banach norm γ_1 ($O(n^{-1/(d+5)})$) and kernel methods ($O(n^{-1/(d+1)})$) (Bach, 2024, Section 7.5, Section 9.4) do not beat the curse of dimensionality, and neither does our setup.

However, in the case of linear latent variables, i.e., under the multiple index model where $f^* = g^*(P^\top x)$ with P a $d \times k$ matrix with $k < d$ and orthonormal columns, the RKHS cannot take advantage of this hypothesis and the rates remain unchanged. In contrast, the neural network can, assuming that g^* has bounded Banach norm, then we only pay the price of the k underlying dimensions and not the full d dimensions (Bach, 2024, Section 9.4). BKerNN also has this property, which is visible by using the simple arguments presented in the discussion in Bach (2024, Section 9.3.5), which show that $\Omega(f) \leq \Omega(g^*)$. Moreover, the optimisation process for BKerNN is much easier than that of neural networks, and our function space is larger, underscoring the attractiveness of BKerNN.

There is also an implicit dependency on the dimension in Theorem 16 through data-dependent terms, namely the variance proxy σ^2 or the expectations in the bound of G_n. We now examine these quantities under two data-generating mechanisms: bounded and subgaussian variables.

Lemma 17 (Analysis of Data-Dependent Terms in Theorem 16) *The following inequalities hold.*

1. *If X is bounded, i.e., $\|X\|^* \leq R$ almost surely, then*

$$\sqrt{\mathbb{E}_X \|X\|^*} \leq \sqrt{R}, \quad \left(\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]} (\|X_i\|^*)^2\right)\right)^{1/4} \leq \sqrt{R}.$$

Moreover, $1 + \sqrt{\|X\|^*}$ is subgaussian with variance proxy $\sigma^2 \leq 1 + \sqrt{R}$.

2. *If X is a vector of subgaussian variables (not necessarily centred or independent) with variance proxy σ_a^2 for component X_a, then*

$$\sqrt{\mathbb{E}_X(\|X\|_2)} \leq \sqrt{6}\left(\sum_{a=1}^d \sigma_a^2\right)^{1/4}, \quad \sqrt{\mathbb{E}_X(\|X\|_\infty)} \leq 4(\log d)^{1/4} \max_{a\in[d]} \sqrt{\sigma_a},$$

$$\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]} \|X_i\|_2^2\right)^{1/4} \leq 4(1+\log(n))^{1/4}\left(\sum_{a=1}^d \sigma_a^2\right)^{1/4},$$

$$\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]} \|X_i\|_\infty\right)^{1/4} \leq 4(1+\log(nd))^{1/4} \max_{a\in[d]} \sqrt{\sigma_a}.$$

Furthermore, $1+\sqrt{\|X\|_2}$ is subgaussian with variance proxy $\sigma^2 \leq (1+\sum_{a=1}^d \sigma_a)^2$, and $1+\sqrt{\|X\|_\infty}$ is subgaussian with variance proxy $\sigma^2 \leq 2+\max_{a\in[d]} \sigma_a^2(1+\sqrt{\log(2d)})^2$.

See the proof in Appendix A.4.4. Note that R usually does not implicitly depend on the dimension in the $\|\cdot\|^* = \|\cdot\|_\infty$ case, and R can typically be $O(d^{1/2})$ in the $\|\cdot\|_2$ case. For the subgaussian mechanism, each σ_a typically does not depend on the dimension.

5 Numerical Experiments

In this section, we present and analyse the properties of BKerNN. The BKerNN implementation in Python is fully compatible with Scikit-learn (Pedregosa et al., 2011), ensuring seamless integration with existing machine learning workflows. The source code, along with all necessary scripts to reproduce the experiments, is available at `https://github.com/BertilleFollain/BKerNN`. We define the scores and other estimators in the section below.

5.1 Introduction to Scores and Competitors

In the experiments below, we use two scores to assess performance. The prediction score is defined as the coefficient of determination, a classical metric in the statistics literature (Wright, 1921), R^2, which ranges from $-\infty$ to 1, where a score of 1 indicates perfect prediction, a score of 0 indicates that the model predicts no better than the mean of the target values, and negative values indicate that the model performs worse than this baseline. Mathematically, the R^2 score is defined as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}, \tag{15}$$

where y_i are the true values, $\hat{y}_i$ are the predicted values, $\bar{y}$ is the mean of the true values, and n is the number of samples.

The feature learning score measures the model's ability to identify and learn the true feature space (1 being the best, 0 the worst). It is computable only when the underlying feature space (in the form of a matrix $P \in \mathbb{R}^{d\times k}$, with k the number of features) is known and relevant only when features are of similar importance, which we have ensured in the experiments below.

Depending on the regularisation type, the estimated feature matrix $\hat{P}$ is computed via singular value decomposition (SVD) for Ω_{feature}, $\Omega_{\text{concave feature}}$ or Ω_{basic} regularisation, or by selecting the top k variables for Ω_{variable} or $\Omega_{\text{concave variable}}$ regularisation. We then compute the projection matrices $\pi_{\hat{P}}$ and π_P and calculate the feature learning error as the normalised Frobenius norm of their difference

$$\pi_{\hat{P}} = \hat{P}(\hat{P}^\top \hat{P})^{-1}\hat{P}^\top \quad \text{and} \quad \pi_P = P(P^\top P)^{-1}P^\top,$$

$$\text{score} = \begin{cases} 1 - \frac{\|\pi_P - \pi_{\hat{P}}\|_F^2}{2k} & \text{if} k \leq \frac{n_{\text{features}}}{2}, \\ 1 - \frac{\|\pi_P - \pi_{\hat{P}}\|_F^2}{2n_{\text{features}} - 2k} & \text{if} k > \frac{n_{\text{features}}}{2}, \end{cases} \tag{16}$$

where the score is 1 if $k = n_{\text{features}}$.

In several experiments, we compare the performance of BKerNN against ReLUNN and BKRR. BKRR refers to Kernel Ridge Regression using the multi-dimensional Brownian kernel $k^{(mdB)}(x, x') = (\|x\| + \|x'\| - \|x - x'\|)/2$. ReLUNN is a simple one-hidden-layer neural network with ReLU activations, trained using batch stochastic gradient descent.

5.2 Experiment 1: Optimisation Procedure, Importance of Positive Homogeneous Kernel

In this experiment, we compare BKerNN with two methods that differ from BKerNN only through the kernel that is used. We wish to illustrate the importance of the homogeneity assumptions discussed in Section 3.2. Specifically, we consider ExpKerNN with the (rescaled) exponential kernel $k^{\exp}(a, b) = e^{-|a-b|/2}$ and GaussianKerNN with the Gaussian kernel $k^{\text{Gaussian}}(a, b) = e^{-|a-b|^2/2}$. Unlike the Brownian kernel used in BKerNN, the exponential and Gaussian kernels are not positively 1-homogeneous.

We trained all three methods on a simulated data set, using cross-validation to select the regularisation parameter λ while keeping other parameters fixed ($m = 100$, basic regularisation, more details are provided in Appendix B.1). The training set consisted of 214 samples and the test set of 1024. The data had $d = 45$ dimensions with $k = 5$ relevant features, and Gaussian additive noise with a standard deviation of 0.5. An orthogonal matrix P of size $d \times d$ was sampled uniformly from the orthogonal group before being truncated to size $d \times k$. The covariates were sampled uniformly from $[-1, 1]^d$, and the target variable y was computed as $y = 2\pi \left| \sum_{a=1}^{k} (P^\top x)_a \right| + \text{noise}$.

We displayed the mean squared error (MSE) on both the training and test sets for the selected λ for each method in Figure 1. While all three methods perform very well on the training set, the test set performance of ExpKerNN and GaussianKerNN is significantly worse compared to BKerNN. This discrepancy is not due to suboptimal regularisation choices, as cross-validation was used to select the best λ for each method.

Instead, the superior test performance of BKerNN underscores its effective optimisation process, avoiding the pitfalls of local minima that seem to trap ExpKerNN and GaussianKerNN. Our observations in Figure 1 strongly support our discussion in Section 3.2 on the critical role of the positive homogeneity of the kernel in ensuring convergence to a global minimum.

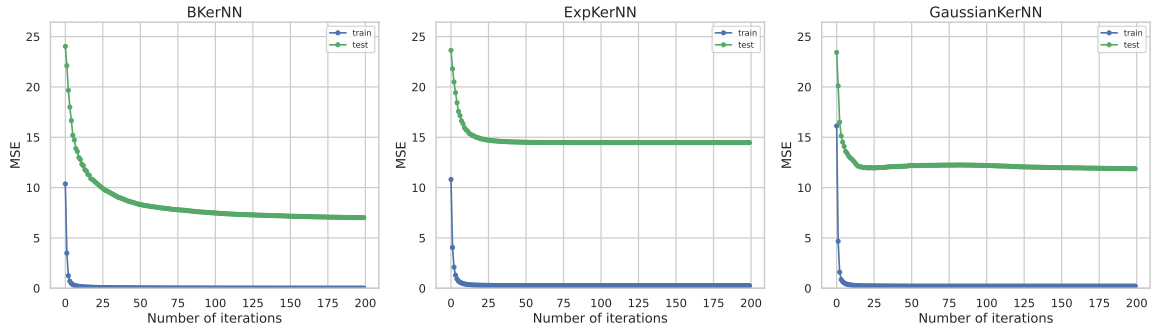

Figure 1: MSE across optimisation procedure for different kernels

5.3 Experiments 2 & 3: Influence of Parameters (Number of Particles m, Regularisation Parameter λ, and Type of Regularisation)

In these experiments, we explore the impact of various parameters on the performance of BKerNN. Detailed descriptions can be found in Appendix B.2, and the results are presented in Figure 2. The R^2 score used to assess performance is described in Equation (15).

5.3.1 Experiment 2

The first two subplots of Figure 2 illustrate the effects of the number of particles m and the regularisation parameter λ while keeping the data generation process consistent. The data set is the same for the two subplots. We used 412 training samples and 1024 test samples, with a data dimensionality of $d = 20$ and $k = 5$ relevant features. The standard deviation of additive Gaussian noise was set to 0.1. The covariates were sampled uniformly from $[-1, 1]^d$. The target variable y was computed using the formula $y = \sum_{a=1}^{k} |2\pi x_a| + \text{noise}$.

Number of Particles (m): the first subplot shows that with too few particles, the estimator struggles to fit the training data, leading to poor performance on the test set. However, beyond a certain threshold, increasing the number of particles does not yield significant improvements in performance.

Regularisation Parameter (λ): the second subplot demonstrates the typical behaviour of a regularised estimator. When λ is too small, the model overfits the training data, resulting in poor test performance. Conversely, when λ is too large, the model underfits, performing poorly on both the training and test sets. Optimal performance on both sets is achieved with an intermediate value of λ.

5.3.2 Experiment 3

The third subplot in Figure 2 examines the influence of the type of regularisation across three distinct data-generating mechanisms: (1) without underlying features, i.e., where all of the data is needed, (2) with few relevant variables, (3) with few relevant features. We used 214 training samples and 1024 test samples, with a data dimensionality of $d = 20$ and $k = 5$ relevant features. The standard deviation of additive Gaussian noise was set to 0.5, and the data set was generated 20 times with different seeds. The covariates were always sampled uniformly on $[-1, 1]^d$ but the response was generated in three different ways. In the

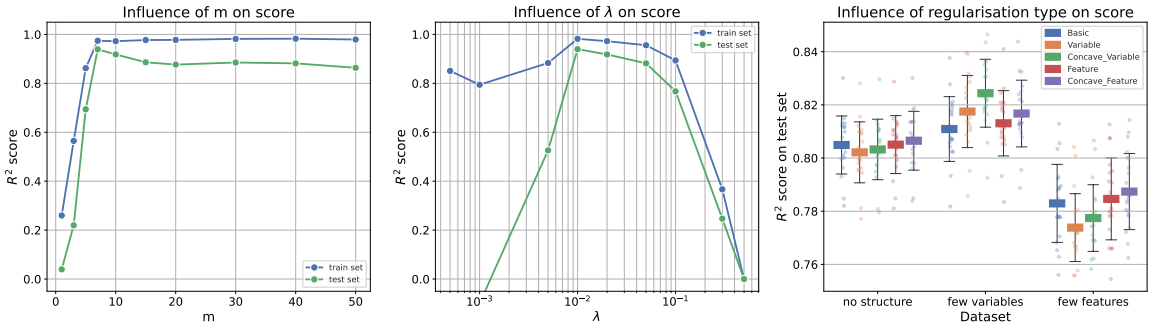

Figure 2: Influence of parameters: left: m, middle: λ, right: type of penalty

"no underlying structure" data set, we had $y = \sum_{a=1}^{d} \sin(X_a) + \text{noise}$. In the "few relevant variables" data set, we had $y = \sum_{a=1}^{k} \sin(x_a) + \text{noise}$. In the "few relevant features data set", we sampled P a $d \times d$ matrix from the orthogonal group uniformly, truncated it to size $d \times k$ and the response was generated as $y = \sum_{a=1}^{k} \sin((P^\top x)_a) + \text{noise}$. The mean and standard deviation of the R^2 score on the test set are reported.

When there is no underlying structure, all regularisers perform somewhat similarly. However, for data sets featuring relevant variables, the Ω_{variable} and $\Omega_{\text{concave variable}}$ regularisations shine, delivering superior performance. Similarly for the Ω_{feature} and $\Omega_{\text{concave feature}}$ regularisations on data sets with few relevant features. Remarkably, for data with underlying structure, the concave versions of both Ω_{variable} and Ω_{feature} regularisations outperform their non-concave counterparts. This demonstrates their superior ability to effectively select relevant information in the data while maintaining strong predictive power.

5.4 Experiment 4: Comparison to Neural Network on 1D Examples, Influence of Number of Particles/Width of Hidden Layer m

In Experiment 4, we compare the learning capabilities of BKerNN against a simple neural network, ReLUNN. We study three distinct functions, corresponding to each row in Figure 3. In all rows, the training set is represented by small black crosses, while the target function is shown in blue. The first two columns depict BKerNN using two different numbers of particles: $m = 1$ and $m = 5$. The last three columns show results for ReLUNN with varying numbers of neurons in the hidden layer: 1, 5, and 32. See Appendix B.3 for more experimental details.

Notably, BKerNN demonstrates great learning capabilities, successfully capturing the functions even with just one particle. Increasing the number of particles (second column) offers minimal additional benefit, underscoring BKerNN's efficiency. In stark contrast, ReLUNN struggles significantly when limited to the same number of hidden neurons as BKerNN's particles. However, once the hidden layer is expanded to 32 neurons, ReLUNN begins to show satisfactory learning capabilities. These results highlight BKerNN's superior efficiency in learning functions with a minimal number of particles, outperforming ReLUNN, which requires a more complex architecture to achieve comparable performance.

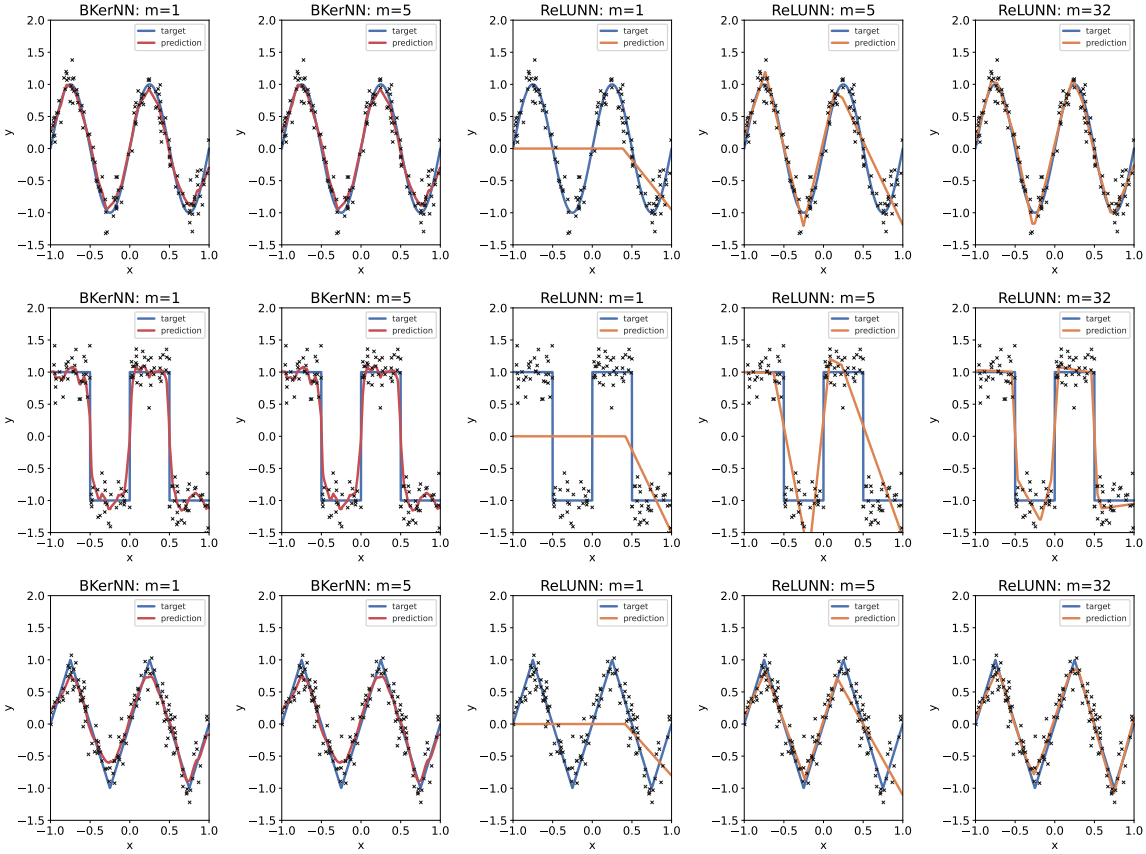

Figure 3: Comparison to neural network on 1D examples

5.5 Experiment 5: Prediction Score and Feature Learning Score Against Growing Dimension and Sample Size, a Comparison of BKerNN with Brownian Kernel Ridge Regression and a ReLU Neural Network

In Experiment 5, we evaluate the performance of BKerNN, BKRR and ReLUNN across varying sample sizes and dimensions on simulated data sets. The estimators are presented in Section 5.1. The R^2 and feature learning score used to assess performance are described in Equations (15) and (16) respectively. The results are presented in Figure 4. For more details about the experiment, see Appendix B.4.

The two subplots on the top row of Figure 4 show the effect of increasing the sample size while keeping the dimension fixed. In the two subplots of the bottom row, the sample size is fixed, and the dimension is increased. For each combination of sample size and dimension, ten data sets were generated. We display the two scores of each method on each data set, as well as the average score across data sets. The feature learning score for BKRR is not defined and, therefore, not displayed. The number of particles (for BKerNN) and hidden neurons (for ReLUNN) is fixed at 50 across all experiments.

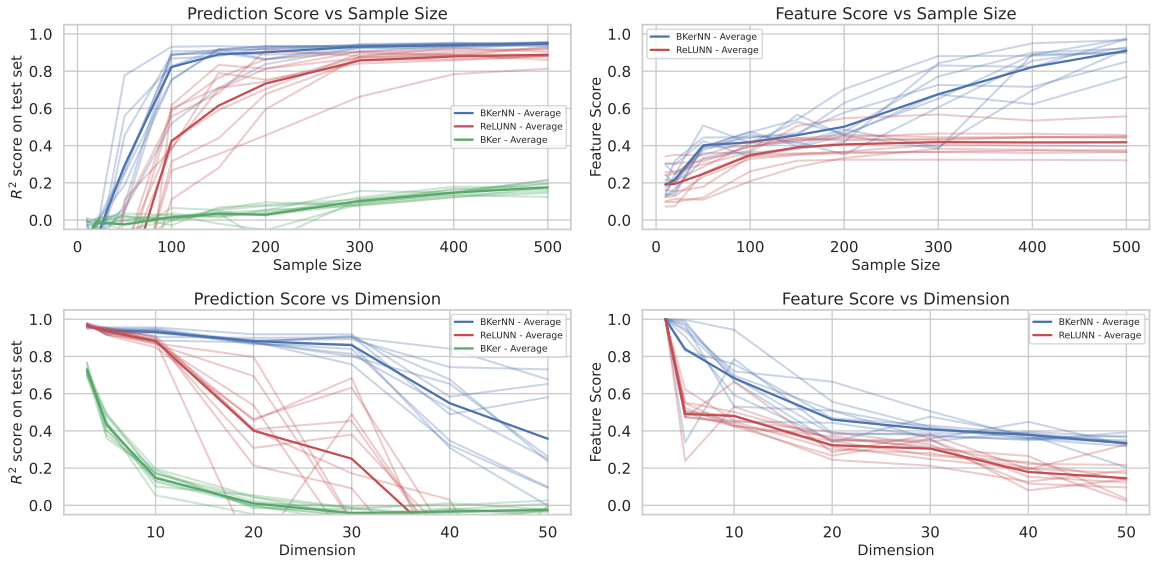

Figure 4: Performance comparison across varying sample sizes and dimensions

For all the data sets, the covariates were uniformly sampled in $[-1,1]^d$, the underlying features matrix P was uniformly sampled from the orthogonal group, then truncated to have $k=3$ relevant features, and the response was set as $y = \left|\sum_{a=1}^{k} \sin\left((P^\top x)_a\right)\right|$.

In the first two subplots, the dimension is fixed at 15. As the sample size increases, we observe improvements in the prediction scores of all three methods. However, the prediction score of BKRR improves at a much slower pace. Both BKerNN and ReLUNN achieve high prediction scores more rapidly, with BKerNN requiring fewer samples to do so. Notably, BKerNN excels in feature learning, effectively capturing the underlying feature space, while ReLUNN fails regardless of the number of samples.

In the last two subplots, where the sample size is fixed at 212, we notice a general decline in performance as the dimension increases. BKRR shows the most rapid deterioration because it cannot learn features, struggling significantly with higher dimensions. In contrast, BKerNN demonstrates remarkable resilience to increasing dimensionality, maintaining better performance compared to the other methods. ReLUNN falls somewhere in between, neither as robust as BKerNN nor as weak as BKRR. Similarly, for the feature learning score, both BKerNN and ReLUNN show decreased performance, but BKerNN is slightly less affected, underscoring its ability to handle high-dimensional data.

5.6 Experiment 6: Comparison on Real Data Sets Between BKerNN, Brownian Kernel Ridge Regression and a ReLU Neural Network

In Experiment 6, we evaluate the R^2 scores, defined in Equation (15), of four methods: BKRR, BKerNN with concave variable regularisation, BKerNN with concave feature regularisation, and ReLUNN, across 17 real-world data sets. These data sets were obtained from the tabular benchmark numerical regression suite via the OpenML platform, as described by Grinsztajn et al. (2022). Each data set was processed to include only numerical

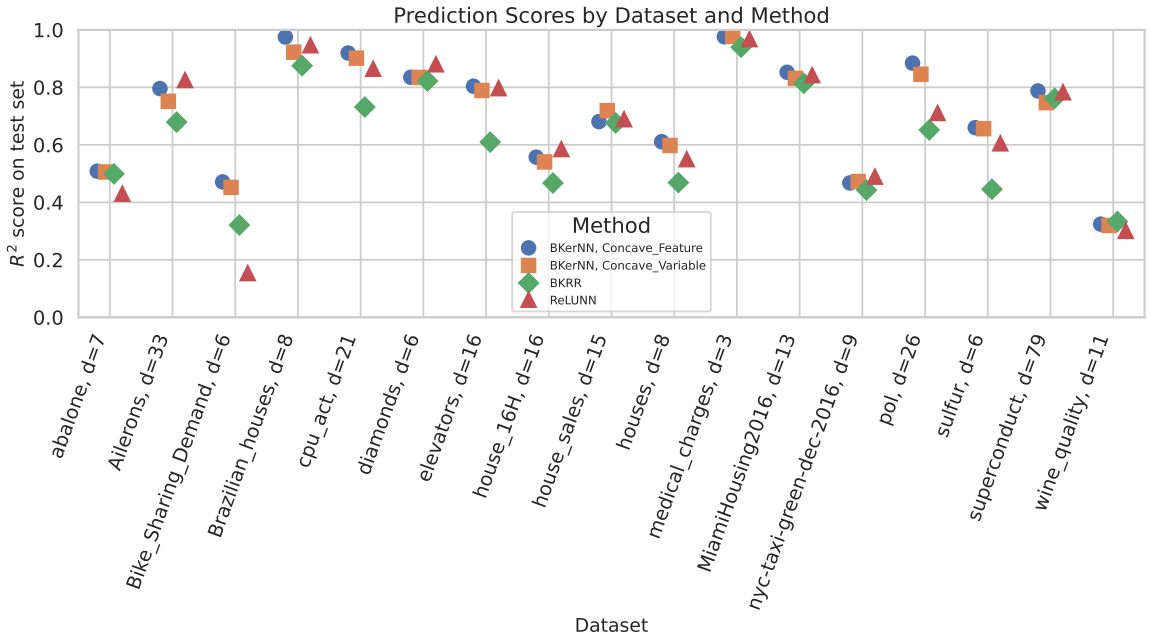

Figure 5: Comparison of R^2 scores on real data sets

variables and rescaled to have centred covariates with standard deviation equal to one. The data sets were uniformly cropped to contain 400 training samples and 100 testing samples, with dimensionality varying across data sets as shown in Figure 5. For both BKERNN and RELUNN, the number of particles or hidden neurons was set to twice the dimension of each data set, while the training parameters were fixed. Details are available in Appendix B.5.

The results indicate that BKRR often performs the worst among all methods. In contrast, BKERNN with concave feature regularisation and RELUNN frequently emerge as the best estimators, performing similarly well on average across the various data sets.

6 Conclusion

To conclude, we have introduced a novel framework for feature learning and function estimation in supervised learning, termed Brownian kernel neural network (BKERNN). By leveraging regularised empirical risk minimisation over averages of Sobolev spaces on one-dimensional projections of the data, we established connections to kernel ridge regression and infinite-width one-hidden layer neural networks. We provide an efficient computational method for BKERNN, emphasising the importance of the positive homogeneity of the Brownian kernel. Through rigorous theoretical analysis, we demonstrated that, in the well-specified setting for subgaussian data, BKERNN achieves convergence of its expected risk to the minimal risk with explicit rates, potentially independent of the data dimension, underscoring the efficacy of our approach. We have extensively discussed the relationship between the space of functions we propose and other classical functions spaces. Numerical experiments across simulated scenarios and real data sets confirm BKERNN's superiority over traditional kernel ridge regression and competitive performance with neural networks

employing ReLU activations, achieved with fewer particles or hidden neurons. Future research directions include the development of more efficient algorithms for the computation of the estimator, improved analysis of the Gaussian complexity, and theoretical investigation of other penalties.

Acknowledgments and Disclosure of Funding

B. Follain would like to thank Adrien Taylor and David Holzmüller for fruitful discussions regarding this work. We acknowledge support from the French government under the management of the Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute). We also acknowledge support from the European Research Council (grant SEQUOIA 724063).

Appendix A. Extra Lemmas and Proofs

In this appendix, we present and/or prove some of the results needed in the main text.

A.1 Proofs of Section 2.3 Lemmas

Here we give the proofs of the lemmas describing characteristics of the function space $\mathcal{F}_\infty$.

A.1.1 Proof of Lemma 3

Proof [Lemma 3] We first check that $\mathcal{F}_\infty$ is a vector space.

Let $f \in \mathcal{F}_\infty$ with $f(\cdot) = c + \int_{\mathcal{S}^{d-1}} g_w(w^\top \cdot)\mathrm{d}\mu(w)$ and $\tau \in \mathbb{R}$ then $\tau f(\cdot) = \tau c + \int_{\mathcal{S}^{d-1}} \tau g_w(w^\top \cdot)\mathrm{d}\mu(w)$ and $\tau g_w \in \mathcal{H}$ hence $\tau f \in \mathcal{F}_\infty$.

Now let $f, \tilde{f} \in \mathcal{F}_\infty$, then $(f+\tilde{f})(\cdot) = c+\tilde{c}+\int_{\mathcal{S}^{d-1}} g_w(w^\top \cdot)\mathrm{d}\mu(w)+\int_{\mathcal{S}^{d-1}} \tilde{g}_w(w^\top \cdot)\mathrm{d}\tilde{\mu}(w) = c+\tilde{c}+\int_{\mathcal{S}^{d-1}}(\mathrm{d}\frac{2\mu(w)}{\mu(w)+\tilde{\mu}(w)}g_w+\mathrm{d}\frac{2\tilde{\mu}(w)}{\mu(w)+\tilde{\mu}(w)}\tilde{g}_w)(w^\top \cdot)\mathrm{d}(\frac{\mu+\tilde{\mu}}{2})(w)$, hence $f+\tilde{f}$ belongs to $\mathcal{F}_\infty$ and $\mathcal{F}_\infty$ is a vector space.

Then, we see as a direct consequence that $\{f \in \mathcal{F}_\infty, \Omega_0(f) < \infty\} = \{f \in \mathcal{F}_\infty, \Omega(f) < \infty\}$ is also a vector space. Let $f, \tilde{f} \in \mathcal{F}_\infty$ with $\Omega_0(f) < \infty$ and $\tau \in \mathbb{R}$. Then, $\Omega(f) = 0 \iff f = 0$ and $\Omega(\tau f) = |\tau|\Omega(f)$. For the triangular inequality, we have

$$\begin{aligned}
\Omega_0(f+\tilde{f}) &= \int_{\mathcal{S}^{d-1}} \left\| \mathrm{d}\frac{2\mu(w)}{\mu(w)+\tilde{\mu}(w)} g_w + \mathrm{d}\frac{2\tilde{\mu}(w)}{\mu(w)+\tilde{\mu}(w)}\tilde{g}_w \right\|_{\mathcal{H}} \mathrm{d}\left(\frac{\mu+\tilde{\mu}}{2}\right)(w) \\
&\leq \int_{\mathcal{S}^{d-1}} \mathrm{d}\frac{2\mu(w)}{\mu(w)+\tilde{\mu}(w)}\|g_w\|_{\mathcal{H}} + \mathrm{d}\frac{2\tilde{\mu}(w)}{\mu(w)+\tilde{\mu}(w)}\|\tilde{g}_w\|_{\mathcal{H}}\mathrm{d}\left(\frac{\mu+\tilde{\mu}}{2}\right)(w) \\
&\leq \Omega_0(f) + \Omega_0(\tilde{f}),
\end{aligned}$$

hence $\Omega(f+\tilde{f}) = \max(f(0)+\tilde{f}(0), \Omega_0(f+\tilde{f})) \leq \Omega(f) + \Omega(\tilde{f})$, which yields that Ω is a norm.

We first check the Hölder continuous property

$$\begin{aligned}
f(x) - f(x') &= c + \int_{\mathcal{S}^{d-1}} g_w(w^\top x)\mathrm{d}\mu(w) - c - \int_{\mathcal{S}^{d-1}} g_w(w^\top x')\mathrm{d}\mu(w) \\
&= \int_{\mathcal{S}^{d-1}} \langle g_w, k_{w^\top x} - k_{w^\top x'}\rangle \mathrm{d}\mu(w) \\
|f(x) - f(x')| &\leq \int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}} \|k_{w^\top x} - k_{w^\top x'}\|_{\mathcal{H}} \mathrm{d}\mu(w) \leq \int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}} \sqrt{|w^\top (x - x')|}\mathrm{d}\mu(w) \\
&\leq \int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}} \sqrt{\|x - x'\|^*}\mathrm{d}\mu(w) \leq \Omega_0(f)\sqrt{\|x - x'\|^*}.
\end{aligned}$$

We then consider the weak derivatives, let $a \in [d]$, then

$$\frac{\partial f}{\partial x_a}(x) = \int_{\mathcal{S}^{d-1}} w_a g'_w(w^\top x)\mathrm{d}\mu(w),$$

which means that

$$\begin{aligned}
\int_{\mathbb{R}^d} \left(\frac{\partial f}{\partial x_a}(x)\right)^2 \mathrm{d}\rho(x) &= \int_{\mathbb{R}^d} \left(\int_{\mathcal{S}^{d-1}} w_a g'_w(w^\top x)\mathrm{d}\mu(w)\right)^2 \mathrm{d}\rho(x) \\
&\leq \int_{\mathbb{R}^d} \left(\int_{\mathcal{S}^{d-1}} |w_a||g'_w(w^\top x)|\mathrm{d}\mu(w)\right)^2 \mathrm{d}\rho(x) \\
&\leq \int_{\mathbb{R}^d} \left(\int_{\mathcal{S}^{d-1}} w_a^2 g'_w(w^\top x)^2 \mathrm{d}\mu(w) \int_{\mathcal{S}^{d-1}} \mathrm{d}\mu(w)\right) \mathrm{d}\rho(x) \\
&= \int_{\mathcal{S}^{d-1}} \int_{\mathbb{R}^d} w_a^2 g'_w(w^\top x)^2 \mathrm{d}\rho(x)\mathrm{d}\mu(w) \\
&= \int_{\mathcal{S}^{d-1}} w_a^2 \int_{\mathbb{R}^d} g'_w(w^\top x)^2 \mathrm{d}\rho(x)\mathrm{d}\mu(w) \\
&\leq \int_{\mathcal{S}^{d-1}} w_a^2 \|g_w\|_{\mathcal{H}} \mathrm{d}\mu(w) \leq \Omega_0(f) \sup_{w \in \mathcal{S}^{d-1}} w_a^2 \\
&< \infty \text{ by equivalence of the norms on } \mathbb{R}^d.
\end{aligned}$$

■

A.1.2 Proof of Lemma 4

Proof [Lemma 4] Let us assume now that we only consider functions f with support on the ball with centre 0, radius R and norm $\|\cdot\|^*$, which we denote $B(0, R)$. Then we can actually consider the functions g_w which define $\mathcal{F}_\infty$ to belong to $\mathcal{H}' := \{g : \mathbb{R} \to \mathbb{R} \dim g(0) = 0, \int_{-R}^{R} (g'(t))^2 \mathrm{d}t\}$, and it is still a RKHS with the same reproducing kernel. Let $f \in \mathcal{F}_\infty, f = c + \int_{\mathcal{S}^{d-1}} g_w(w^\top \cdot)\mathrm{d}\mu(w)$. Then, because we have restrained on the ball and we are continuous, f is necessarily in $L_1(B(0, R))$ (set of integrable functions) and in $L_2(B(0, R))$. It has a Fourier decomposition, with

$$f(x) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{f}(\omega) e^{i\omega^\top x} \mathrm{d}\omega,$$

and then we have

$$\Omega_0(f) \leq \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} |\hat{f}(\omega)| \Omega_0(e^{i\omega^\top x}) \mathrm{d}\omega$$

and we can then study $\Omega_0(e^{i\omega^\top x})$.

We have $e^{i\omega^\top x} = g_\omega\big(\frac{\omega}{\|\omega\|}^\top x\big)$ with $g_\omega : t \in [-R, R] \to e^{it\|\omega\|}$ which belongs to (the complex version of) $\mathcal{H}$, with $\|g_\omega\|_{\mathcal{H}} = \sqrt{\int_{-R}^{R} \|\omega\|^2 |e^{it\|\omega\|}|^2 \mathrm{d}t} \leq \sqrt{2R}\|\omega\|$.

This yields

$$\Omega_0(f) \leq \frac{\sqrt{2R}}{(2\pi)^d} \int_{\mathbb{R}^d} |\hat{f}(\omega)| \cdot \|\omega\| \mathrm{d}\omega.$$

■

A.2 Proofs of Section 2.4 Lemmas

In this section we present the proof of lemmas used to transform the optimisation problem defining BKerNN.

A.2.1 Proof of Lemma 5

Proof [Lemma 5] Our goal is to transform Equation (5). We begin with the following trick for the m particles setting

$$\frac{1}{m}\sum_{j=1}^{m} \|g_j\|_{\mathcal{H}} = \inf_{\beta \in \mathbb{R}_+^m} \frac{1}{2m} \sum_{j=1}^{m} \left(\frac{\|g_j\|_{\mathcal{H}}^2}{\beta_j} + \beta_j \right).$$

Fix $(w_j)_{j\in[m]}$ and $(\beta_j)_{j\in[m]}$ in Equation (5), yielding the following minimisation problem on the functions $(g_j)_{j\in[m]}$

$$\min_{c\in\mathbb{R}, g_1,\dots,g_m \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, c + \frac{1}{m}\sum_{j=1}^{m} g_j(w_j^\top x_i)) + \frac{\lambda}{2}\frac{1}{m}\sum_{j=1}^{m} \frac{\|g_j\|_{\mathcal{H}}^2}{\beta_j}. \tag{17}$$

Using the representer theorem (Schölkopf et al., 2001), we express each $x \to g_j(w_j^\top x)$ as

$$x \to \sum_{i=1}^{n} \alpha_i^{(j)} k^{(B)}(w_j^\top x_i, w_j^\top x),$$

which leads to

$$\|g_j\|_{\mathcal{H}}^2 = \sum_{i,i'=1}^{n} \alpha_i^{(j)} \alpha_{i'}^{(j)} k^{(B)}(w_j^\top x_i, w_j^\top x_{i'}).$$

Rewriting the norm and evaluation in kernel form with $K_{i,i'}^{(w_j)} = k^{(B)}(w_j^\top x_i, w_j^\top x_{i'})$, we obtain

$$\|g_j\|_{\mathcal{H}}^2 = (\alpha^{(j)})^\top K^{(w_j)} \alpha^{(j)},$$

and

$$g_j(w_j^\top x_i) = (K^{(w_j)}\alpha^{(j)})_i.$$

Thus, we transform Equation (17) into

$$\min_{c\in\mathbb{R},\alpha^{(1)},\dots,\alpha^{(m)}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \ell(y_i, \frac{1}{m}\sum_{j=1}^m (K^{(w_j)}\alpha^{(j)})_i + c) + \frac{\lambda}{2}\frac{1}{m}\sum_{j=1}^m \frac{(\alpha^{(j)})^\top K^{(w_j)}\alpha^{(j)}}{\beta_j}.$$

We show that minimisation is attained for vectors $\alpha^{(j)}$ equal to $\beta_j\alpha$ for a single vector α. Consider the convex problem

$$\min_{\alpha^{(1)},\dots,\alpha^{(m)}\in\mathbb{R}^d} \frac{1}{2}\frac{1}{m}\sum_{j=1}^m \frac{(\alpha^{(j)})^\top K^{(w_j)}\alpha^{(j)}}{\beta_j},$$

subject to $\frac{1}{m}\sum_{j=1}^m K^{(w_j)}\alpha^{(j)} = z$ where $z\in\mathbb{R}^d$. We define the Lagrangian

$$\mathcal{L}(\alpha^{(1)},\cdots,\alpha^{(m)},\alpha) = \frac{1}{2}\frac{1}{m}\sum_{j=1}^m \frac{(\alpha^{(j)})^\top K^{(w_j)}\alpha^{(j)}}{\beta_j} + \alpha^\top\left(z - \frac{1}{m}\sum_j K^{(w_j)}\alpha^{(j)}\right).$$

By taking the differential of $\mathcal{L}$ with respect to $\alpha^{(j)}$ at the optimum, we get

$$\frac{\partial\mathcal{L}}{\partial\alpha^{(j)}} = \frac{1}{m}K^{(w_j)}\left(\frac{\alpha^{(j)}}{\beta_j} - \alpha\right) = 0.$$

The differential with respect to α yields that at the optimum, the constraint is verified, i.e., $z = \frac{1}{m}\sum_j K^{(w_j)}\alpha^{(j)}$. We note that for $\alpha^{(j)} = \beta_j\alpha$, all equations are satisfied, yielding the desired result.

We can then write Equation (5) as

$$\min_{w_1,\dots,w_m\in\mathbb{R}^d, c\in\mathbb{R}, \beta\in\mathbb{R}_+^m, \alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^n \ell(y_i,(K\alpha)_i + c) + \frac{\lambda}{2}\alpha^\top K\alpha + \frac{\lambda}{2}\frac{1}{m}\sum_{j=1}^m \beta_j,$$

with the constraints $\forall j\in[m], w_j\in\mathcal{S}^{d-1}$, and $K = \frac{1}{m}\sum_{j=1}^m \beta_j K^{(w_j)}$.

We notice that $\beta_j K^{(w_j)} = K^{(\beta_j w_j)}$ due to the positive homogeneity of the Brownian kernel. We therefore introduce the change of variable $\beta_j w_j = \tilde{w}_j$

$$\min_{\tilde{w},\dots,\tilde{w}_m\in\mathbb{R}^d, c\in\mathbb{R}, \beta\in\mathbb{R}_+^m, \alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^n \ell(y_i,(K\alpha)_i + c) + \frac{\lambda}{2}\alpha^\top K\alpha + \frac{\lambda}{2}\frac{1}{m}\sum_{j=1}^m \|\tilde{w}_j\|,$$

with $K = \frac{1}{m}\sum_{j=1}^m K^{(\tilde{w}_j)}$ and no constraint on the norm of $\tilde{w}_j$. For ease of exposition in the main text, we replace $\tilde{w}$ by w. ∎

A.2.2 Proof of Lemma 6

Proof [Lemma 6] The proof follows the same steps as the proof of Lemma 5, systematically replacing any $\frac{1}{m}\sum_{j=1}^{m}$ with the appropriate integral over $\mathcal{S}^{d-1}$ with respect to measure μ. Before the change of variables, the problem is

$$\min_{\mu\in\mathcal{P}(\mathcal{S}^{d-1}),c\in\mathbb{R},(\beta_w)_w\in\mathbb{R}_+^{\mathcal{S}^{d-1}},\alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n}\ell(y_i,(K\alpha)_i+c)+\frac{\lambda}{2}\alpha^\top K\alpha+\frac{\lambda}{2}\int_{\mathcal{S}^{d-1}}\beta_w \mathrm{d}\mu(w),$$

where $K=\int_{\mathcal{S}^{d-1}}\beta_w K^{(w)}\,d\mu(w)=\int_{\mathcal{S}^{d-1}}K^{(\beta_w w)}\,d\mu(w)$. The change of variables $\beta_w w=\tilde{w}$ transforms the problem into

$$\min_{(\beta_w)_w\in\mathbb{R}_+^{\mathcal{S}^{d-1}},\nu\in\mathcal{P}(\{\beta_w w,w\in\mathcal{S}^{d-1}\}),c\in\mathbb{R},\alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n}\ell(y_i,(K\alpha)_i+c)+\frac{\lambda}{2}\alpha^\top K\alpha+\frac{\lambda}{2}\int_{\mathbb{R}^d}\|\tilde{w}\|\mathrm{d}\nu(\tilde{w}),$$

with $K=\int_{\mathbb{R}^d}K^{(\tilde{w})}\mathrm{d}\nu(\tilde{w})$. We can consider the integral over $\mathbb{R}^d$ instead of $\{\beta_w w,w\in\mathcal{S}^{d-1}\}$ by extending ν with $\nu(\mathbb{R}^d\setminus\{\beta_w w,w\in\mathcal{S}^{d-1}\})=0$. This is equivalent to considering the minimum over $\nu\in\mathcal{P}(\mathbb{R}^d)$ instead of the minimum over $(\beta_w)_{w\in\mathcal{S}^{d-1}}\in\mathbb{R}_+$ and $\nu\in\mathcal{P}(\{\beta_w w,w\in\mathcal{S}^{d-1}\})$.

The first minimum is smaller as it is considered over a larger space, but they are equal because both the norm $\|\cdot\|$ and the kernel K are positively homogeneous. Hence, the problem finally becomes

$$\min_{\nu\in\mathcal{P}(\mathbb{R}^d),c\in\mathbb{R},\alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n}\ell(y_i,(K\alpha)_i+c)+\frac{\lambda}{2}\alpha^\top K\alpha+\frac{\lambda}{2}\int_{\mathbb{R}^d}\|\tilde{w}\|\mathrm{d}\nu(\tilde{w}),$$

with $K=\int_{\mathbb{R}^d}K^{(\tilde{w})}\mathrm{d}\nu(\tilde{w})$. Learning an optimal ν yields an optimal μ by taking $\mathrm{d}\mu(w)=\mathrm{d}\nu(\{\tilde{w}\in\mathbb{R}^d\mid\tilde{w}/\|\tilde{w}\|=w\})$. For ease of exposition in the main text, we replace $\tilde{w}$ by w. ■

A.3 Proofs of Section 3.1 Lemmas

In this section, we provide the proofs of the lemmas used to compute the estimator.

A.3.1 Proof of Lemma 7

Proof [Lemma 7] For a fixed α, the optimal c is given by $c=\frac{\mathbb{1}^\top Y}{n}-\frac{\mathbb{1}^\top K\alpha}{n}$. Substituting this back into the objective function, we obtain

$$\min_{\alpha\in\mathbb{R}^n}\frac{1}{2n}\|\Pi Y-\Pi K\alpha\|_2^2+\frac{\lambda}{2}\alpha^\top K\alpha,$$

which is minimised for α satisfying $(K\Pi K+n\lambda K)\alpha=K\Pi Y$. We can further simplify this by observing that if $(\Pi K+n\lambda I)\alpha=\Pi Y$, then the previous condition is satisfied.

From the equation $n\lambda\alpha=\Pi Y-\Pi K\alpha$, we can deduce that $\Pi\alpha=\alpha$ because $\Pi^2=\Pi$. Therefore, we can express α as $\alpha=\Pi\tilde{\alpha}$. Substituting this change of variable into the original problem, we define $\tilde{K}:=\Pi K\Pi$ and $\tilde{Y}:=\Pi Y$, transforming the problem into

$$\min_{\tilde{\alpha}\in\mathbb{R}^n}\frac{1}{2n}\|\tilde{Y}-\tilde{K}\tilde{\alpha}\|_2^2+\frac{\lambda}{2}\tilde{\alpha}^\top\tilde{K}\tilde{\alpha}.$$

This is a standard kernel ridge regression problem (noting that $\tilde{K}$ is still a valid kernel matrix), for which the solution is known to be $\tilde{\alpha} = (\tilde{K} + n\lambda I)^{-1}\tilde{Y}$. We also have $\Pi\tilde{\alpha} = \tilde{\alpha}$, implying $\alpha = \tilde{\alpha}$ because one can show that $\mathbb{1}^\top\tilde{\alpha} = 0$. To see why, note that $\mathbb{1}^\top\tilde{\alpha} = \langle\mathbb{1}, \tilde{\alpha}\rangle = \langle(\tilde{K} + n\lambda I)^{-1}\mathbb{1}, \tilde{Y}\rangle$. Since $(\tilde{K} + n\lambda I)^{-1}\mathbb{1}$ is proportional to $\mathbb{1}$ (as $\mathbb{1}$ is an eigenvector of $\tilde{K} + n\lambda I$ and its inverse), and $\langle\tilde{Y}, \mathbb{1}\rangle = 0$, we obtain the desired result.

Finally, we verify the optimal condition $(K\Pi K + n\lambda K)\alpha = K\Pi Y$. Given $(\Pi K\Pi + n\lambda I)\tilde{\alpha} = \Pi Y$ by definition, multiplying by K yields $(K\Pi K\Pi + n\lambda K)\tilde{\alpha} = K\Pi Y$. Since $\tilde{\alpha} = \alpha = \Pi\alpha$, the desired result follows. ■

A.3.2 Proof of Lemma 8

Proof [Lemma 8] First, we compute the derivative of $G = \frac{\lambda}{2}\tilde{Y}^\top(\tilde{K} + \lambda nI)^{-1}\tilde{Y}$ with respect to w_j

$$\begin{aligned}\frac{\partial G}{\partial w_j} &= \sum_{i,i'=1}^{n} \frac{\partial G}{\partial K_{i,i'}}\frac{\partial K_{i,i'}}{\partial w_j} \\ &= \frac{1}{m}\sum_{i,i'=1}^{n} \frac{\partial G}{\partial K_{i,i'}}\frac{\Big(\operatorname{sign}(w_j^\top x_i)x_i + \operatorname{sign}(w_j^\top x_{i'})x_{i'} - \operatorname{sign}(w_j^\top(x_i - x_{i'}))(x_i - x_{i'})\Big)}{2}.\end{aligned} \tag{18}$$

We know that

$$\frac{\partial G}{\partial(\tilde{K} + \lambda nI)} = -\frac{\lambda}{2}(\tilde{K} + \lambda nI)^{-1}\tilde{Y}\tilde{Y}^\top(\tilde{K} + \lambda nI)^{-1},$$

thus

$$\begin{aligned}\frac{\partial G}{\partial K_{i,i'}} &= \sum_{l,k}\frac{\partial G}{\partial(\tilde{K} + \lambda nI)_{l,k}}\frac{\partial(\Pi K\Pi + \lambda nI)_{l,k}}{\partial K_{i,i'}} \\ &= \sum_{l,k} -\frac{\lambda}{2}((\tilde{K} + \lambda nI)^{-1}\tilde{Y}\tilde{Y}^\top(\tilde{K} + \lambda nI)^{-1})_{l,k}\Pi_{l,i}\Pi_{i',k} \\ &= -\frac{\lambda}{2}\Big(\Pi(\tilde{K} + \lambda nI)^{-1}\tilde{Y}\tilde{Y}^\top(\tilde{K} + \lambda nI)^{-1}\Pi\Big)_{i,i'} \\ &= -\frac{\lambda}{2}(\Pi(\tilde{K} + \lambda nI)^{-1}\tilde{Y})_i(\Pi(\tilde{K} + \lambda nI)^{-1}\tilde{Y})_{i'}.\end{aligned}$$

Substituting this back into Equation (18) and introducing $S_j \in \mathbb{R}^{n\times n}$ with $(S_j)_{i,i'} = (\operatorname{sign}(w_j^\top x_i)x_i + \operatorname{sign}(w_j^\top x_{i'})x_{i'} - \operatorname{sign}(w_j^\top(x_i - x_{i'}))(x_i - x_{i'}))/2$, we get

$$\begin{aligned}\frac{\partial G}{\partial w_j} &= -\frac{\lambda}{2m}\sum_{i,i'=1}^{n}(\Pi(\tilde{K} + \lambda nI)^{-1}\tilde{Y})_i(\Pi(\tilde{K} + \lambda nI)^{-1}\tilde{Y})_{i'}(S_j)_{i,i'} \\ &= -\frac{\lambda}{2m}\operatorname{tr}\Big((\Pi(\tilde{K} + \lambda nI)^{-1}\tilde{Y})^\top S_j(\Pi(\tilde{K} + \lambda nI)^{-1}\tilde{Y})\Big) \\ &= -\frac{\lambda}{2m}\operatorname{tr}\Big(((\tilde{K} + \lambda nI)^{-1}\tilde{Y})^\top\Pi S_j\Pi((\tilde{K} + \lambda nI)^{-1}\tilde{Y})\Big).\end{aligned}$$

This implies that we can replace $(S_j)_{i,i'}$ with the i-th, i'-th component of any matrix with the same centred version, such as $\tilde{S}_j$ where $(\tilde{S}_j)_{i,i'} = -\text{sign}(w_j^\top(x_i - x_i'))(x_i - x_i')$, yielding the desired result. ■

A.3.3 Proof of Lemma 9

Proof [Lemma 9] We consider each penalty separately.

1. For $\Omega_{\text{basic}}(W) = \frac{1}{2m}\sum_{j=1}^m \|w_j\|$, the penalty corresponds to a group Lasso penalty on $W \in \mathbb{R}^{d\times m}$, where the groups are the columns. The proximal operator is given by:

$$\big(\text{prox}_{\lambda\gamma\Omega}(W)\big)_j = \left(1 - \frac{\lambda\gamma}{2m}\frac{1}{\|w_j\|}\right)_+ w_j,$$

   as detailed in (Bach et al., 2012, Section 3.3).

2. For $\Omega_{\text{variable}}(W) = \frac{1}{2}\sum_{a=1}^d \big(\frac{1}{m}\sum_{j=1}^m |(w_j)_a|^2\big)^{1/2}$, this is a group Lasso setting where the groups are the rows of W. The proximal operator is:

$$(\text{prox}_{\lambda\gamma\Omega}(w))^{(a)} = \left(1 - \frac{\lambda\gamma}{2\sqrt{m}}\frac{1}{\|W^{(a)}\|_2}\right)_+ W^{(a)},$$

   also found in Bach et al. (2012, Section 3.3).

3. For $\Omega_{\text{feature}}(W) = \frac{1}{2}\operatorname{tr}\big(\big(\frac{1}{m}\sum_{j=1}^m w_j w_j^\top\big)^{1/2}\big)$, this penalty corresponds to a Lasso penalty on the singular values. Given $W = USV^\top$ (SVD), we have:

$$\text{prox}_{\lambda\gamma\Omega}(W) = U\tilde{S}V^\top \quad \text{with} \quad \tilde{S} = \left(1 - \frac{\lambda\gamma}{2\sqrt{m}|S|}\right)_+ S,$$

   using results from Bach et al. (2012, Section 3.3).

4. For $\Omega_{\text{concave variable}}(W) = \frac{1}{2s}\sum_{a=1}^d \log\big(1 + s\big(\frac{1}{m}\sum_{j=1}^m |(w_j)_a|^2\big)^{1/2}\big)$, the loss is separable along the d dimensions. Considering each $W^{(a)}$ separately, we compute the proximal operator:

$$\text{prox}_{\frac{\lambda\gamma}{2s}\log(1+\frac{s}{\sqrt{m}}\|\cdot\|_2)}(W^{(a)}) = \min_{u^{(a)}\in\mathbb{R}^m} \frac{1}{2}\|W^{(a)} - u^{(a)}\|_2^2 + \frac{\lambda\gamma}{2s}\log(1 + \frac{s}{\sqrt{m}}\|u^{(a)}\|_2).$$

   The subgradients of $\mathcal{L}(u^{(a)}) := \frac{1}{2}\|W^{(a)} - u^{(a)}\|_2^2 + \frac{\lambda\gamma}{2s}\log(1 + \frac{s}{\sqrt{m}}\|u^{(a)}\|_2)$ are:

$$\frac{\partial\mathcal{L}}{\partial u^{(a)}} = -(W^{(a)} - u^{(a)}) + \frac{\lambda\gamma}{2s}\frac{s}{\sqrt{m}}\frac{1}{1 + \frac{s}{\sqrt{m}}\|u^{(a)}\|_2}v^{(a)},$$

   where $\|v^{(a)}\|_2 \le 1$ if $u^{(a)} = 0$, and otherwise $v^{(a)} = u^{(a)}/\|u^{(a)}\|_2$.
   For $u^{(a)} \neq 0$, there is a scalar $c \in \mathbb{R}^+$ such that $u^{(a)} = cW^{(a)}$, yielding:

$$c\left(1 + \frac{\lambda\gamma}{2\sqrt{m}}\frac{1}{c\|W^{(a)}\|_2}\frac{1}{1 + \frac{sc}{\sqrt{m}}\|W^{(a)}\|_2}\right) = 1.$$

This is a second-order polynomial in c that can be solved explicitly. The determinant Δ is

$$\Delta = \big(1 - \frac{s}{\sqrt{m}}\|W^{(a)}\|_2\big)^2 - 4\left(\frac{\lambda\gamma}{2\sqrt{m}}\frac{1}{\|W^{(a)}\|_2} - 1\right)\frac{s}{\sqrt{m}}\|W^{(a)}\|_2.$$

When $\Delta \leq 0$, the proximal operator is $u^{(a)} = 0$. Otherwise, it suffices to compare the two possible values of c and choose the one for which $\mathcal{L}$ is the smallest.

5. For $\Omega_{\text{concave feature}}(W) = \frac{1}{2s}\sum_{a=1}^{d}\log\big(1 + \frac{s}{\sqrt{m}}\sigma_a(w_1,\dots,w_n)\big)$, we combine the results of the third and fourth items above. The proximal operator is

$$\operatorname{prox}_{\lambda\gamma\Omega}(W) = U\tilde{S}V^\top,$$

where $\tilde{S}$ is obtained by replacing all $\|W^{(a)}\|_2$ by σ_a in the computations of the proximal of $\Omega_{\text{concave variable}}$.

■

A.4 Extra Lemma and Proofs Related to Section 4 Except Section 4.2

Here we provided the proofs of the lemmas used to bound the Gaussian complexity.

A.4.1 Proof of Lemma 11

Proof [Lemma 11] Recall that

$$G_n(\{f \in \mathcal{F}_\infty, \Omega(f) \leq D\}) = \mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega_0(f)\leq D, c\leq D}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i f(x_i)\right).$$

We start by considering the expectation over ε only. Using the definitions, we obtain

$$\begin{aligned}
&\mathbb{E}_\varepsilon\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i f(x_i)\right)\\
&\quad= \mathbb{E}_\varepsilon\left(\sup_{|c|\leq D,\int_{\mathcal{S}^{d-1}}\|g_w\|_{\mathcal{H}}\mathrm{d}\mu(w)\leq D}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\left(c + \int_{\mathcal{S}^{d-1}} g_w(w^\top x_i)\mathrm{d}\mu(w)\right)\right)\\
&\quad= \mathbb{E}_\varepsilon\left(D\frac{1}{n}\left|\sum_{i=1}^{n}\varepsilon_i\right| + \sup_{\int_{\mathcal{S}^{d-1}}\|g_w\|_{\mathcal{H}}\mathrm{d}\mu(w)\leq D}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\int_{\mathcal{S}^{d-1}}\langle g_w, k^{(B)}_{w^\top x_i}\rangle\mathrm{d}\mu(w)\right)\\
&\quad\leq D\frac{1}{\sqrt{n}} + \mathbb{E}_\varepsilon\left(\sup_{\int_{\mathcal{S}^{d-1}}\|g_w\|_{\mathcal{H}}\mathrm{d}\mu(w)\leq D}\int_{\mathcal{S}^{d-1}}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\langle g_w, k^{(B)}_{w^\top x_i}\rangle\mathrm{d}\mu(w)\right)
\end{aligned}$$

For the second term of the equation right above, we then have equality to

$$
\begin{aligned}
&= \mathbb{E}_\varepsilon\Bigg(\sup_{\int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}} \mathrm{d}\mu(w) \le D} \int_{\mathcal{S}^{d-1}} \Big| \frac{1}{n} \sum_{i=1}^n \varepsilon_i \langle g_w, k^{(B)}_{w^\top x_i} \rangle \Big| \mathrm{d}\mu(w) \Bigg) \\
&= \mathbb{E}_\varepsilon\Bigg(\sup_{\int_{\mathcal{S}^{d-1}} \|g_w\|_{\mathcal{H}} \mathrm{d}\mu(w) \le D} \sup_{w \in \mathcal{S}^{d-1}} \Big| \frac{1}{n} \sum_{i=1}^n \varepsilon_i \langle g_w, k^{(B)}_{w^\top x_i} \rangle \Big| \Bigg) \\
&= \mathbb{E}_\varepsilon\Bigg(\sup_{w \in \mathcal{S}^{d-1}, \|g\|_{\mathcal{H}} \le D} \Big| \frac{1}{n} \sum_{i=1}^n \varepsilon_i \langle g, k^{(B)}_{w^\top x_i} \rangle \Big| \Bigg) \\
&= \mathbb{E}_\varepsilon\Bigg(\sup_{w \in \mathcal{S}^{d-1}, \|g\|_{\mathcal{H}} \le D} \Big| \langle g, \frac{1}{n} \sum_{i=1}^n \varepsilon_i k^{(B)}_{w^\top x_i} \rangle \Big| \Bigg) = \mathbb{E}_\varepsilon\Bigg(\sup_{w \in \mathcal{S}^{d-1}, \|g\|_{\mathcal{H}} \le D} \langle g, \frac{1}{n} \sum_{i=1}^n \varepsilon_i k^{(B)}_{w^\top x_i} \rangle \Bigg) \\
&= D\mathbb{E}_\varepsilon\Bigg(\sup_{w \in \mathcal{S}^{d-1}, \|g\|_{\mathcal{H}} \le 1} \frac{1}{n} \sum_{i=1}^n \varepsilon_i \langle g, k^{(B)}_{w^\top x_i} \rangle \Bigg) = D\mathbb{E}_\varepsilon\Bigg(\sup_{w \in \mathcal{S}^{d-1}, \|g\|_{\mathcal{H}} \le 1} \frac{1}{n} \sum_{i=1}^n \varepsilon_i g(w^\top x_i) \Bigg).
\end{aligned}
$$

Taking the expectation over the data set on both sides yields the desired result. ■

A.4.2 Lemma 18 and its Proof

This lemma provides an explicit formula for computing the supremum over functions within the unit ball of $\mathcal{H}$, which we can then use for the calculation of Gaussian complexity.

Lemma 18 (Optimal g in Gaussian Complexity) *For any data set $(x_1, \ldots, x_n)$ and $w \in \mathbb{R}^d$, with $K^{(w)} \in \mathbb{R}^{n \times n}$ the kernel matrix of kernel $k^{(B)}$ with data $(w^\top x_1, \ldots, w^\top x_n)$ and $\varepsilon \in \mathbb{R}^n$,*

$$
\sup_{\|g\|_{\mathcal{H}} \le 1} \frac{1}{n} \sum_{i=1}^n \varepsilon_i g(w^\top x_i) = \frac{1}{n} \sqrt{\varepsilon^\top K^{(w)} \varepsilon}
$$

Proof [Lemma 18] By applying the definitions, we obtain

$$
\begin{aligned}
\sup_{\|g\|_{\mathcal{H}} \le 1} \frac{1}{n} \sum_{i=1}^n \varepsilon_i g(w^\top x_i) &= \sup_{\|g\|_{\mathcal{H}} \le 1} \frac{1}{n} \sum_{i=1}^n \varepsilon_i \langle g, k^{(B)}_{w^\top x_i} \rangle = \sup_{\|g\|_{\mathcal{H}} \le 1} \langle g, \frac{1}{n} \sum_{i=1}^n \varepsilon_i k^{(B)}_{w^\top x_i} \rangle \\
&= \frac{1}{n} \left\langle \frac{\sum_{i=1}^n \varepsilon_i k^{(B)}_{w^\top x_i}}{\| \sum_{i=1}^n \varepsilon_i k^{(B)}_{w^\top x_i} \|_{\mathcal{H}}}, \sum_{j=1}^n \varepsilon_j k^{(B)}_{w^\top x_j} \right\rangle \\
&= \frac{1}{n} \| \sum_{i=1}^n \varepsilon_i k^{(B)}_{w^\top x_i} \|_{\mathcal{H}} = \frac{1}{n} \sqrt{\varepsilon^\top K^{(w)} \varepsilon},
\end{aligned}
$$

which is the desired result. ■

A.4.3 Proof of Lemma 13

Proof [Lemma 13] Define g_ζ such that $g_\zeta(0) = 0$ and $g'_\zeta(x) = \min(|g'(x)|, 1/\zeta)\operatorname{sign}(g'(x))$. Note that $\|g'_\zeta\|_\infty \leq \frac{1}{\zeta}$, thus g_ζ is $\frac{1}{\zeta}$-Lipschitz. Additionally, for any $a \in \mathbb{R}$,

$$\begin{aligned}
|g_\zeta(a) - g(a)| &= \left| \int_0^a \big(g'_\zeta(t) - g'(t)\big)\,\mathrm{d}t \right| \\
&\leq \int_0^a \big|g'_\zeta(t) - g'(t)\big|\,\mathrm{d}t \leq \int_{-\infty}^{+\infty} \mathbb{1}_{|g'(t)|\geq 1/\zeta}\big(|g'(t)| - 1/\zeta\big)\,\mathrm{d}t \\
&\leq \sqrt{\int_{-\infty}^{+\infty} \mathbb{1}_{|g'(t)|\geq 1/\zeta}\,\mathrm{d}t \cdot \int_{-\infty}^{+\infty} \big(|g'(t)| - 1/\zeta\big)^2\,\mathrm{d}t} \\
&\leq \sqrt{\int_{-\infty}^{+\infty} \mathbb{1}_{|g'(t)|\geq 1/\zeta}\,\mathrm{d}t} \quad \text{since} \quad \int_{-\infty}^{+\infty} \big(g'(t)\big)^2\,\mathrm{d}t \leq 1 \\
&\leq \zeta \quad \text{since} \quad \int_{-\infty}^{+\infty} \mathbb{1}_{|g'(t)|\geq 1/\zeta}\frac{1}{\zeta^2}\,\mathrm{d}t \leq \int_{-\infty}^{+\infty} \mathbb{1}_{|g'(t)|\geq 1/\zeta}\big(g'(t)\big)^2\,\mathrm{d}t \leq 1,
\end{aligned}$$

yielding the desired result. ■

A.4.4 Proof of Lemma 17

Proof [Lemma 17] We begin with the bounded case. The bounds on the expectations are clearly valid. Then, since $1 + \sqrt{\|X\|^*}$ is a bounded variable, it is necessarily subgaussian with a variance proxy bounded by $\frac{(1+\sqrt{R})^2}{2\log(2)} \leq (1+\sqrt{R})^2$ (Vershynin, 2018, Proposition 2.5.2 (iv)).

Next, we consider the subgaussian case. Using the Cauchy-Schwarz inequality, we handle the case where $\|\cdot\|^* = \|\cdot\|_2$ using Vershynin (2018, Proposition 2.5.2)

$$\sqrt{\mathbb{E}_X(\|X\|_2)} \leq \big(\mathbb{E}_X(\|X\|_2^2)\big)^{1/4} \leq \sqrt{6}\big(\sum_{a=1}^d \sigma_a^2\big)^{1/4}.$$

For the $\|\cdot\|_\infty$ case, applying Vershynin (2018, Exercise 2.5.10) with the constant made explicit yields the desired result.

For the second expectation with $\|\cdot\|^* = \|\cdot\|_2$, we have

$$\begin{aligned}
\mathbb{E}_{\mathcal{D}_n}\big(\max_{i\in[n]}\|X_i\|_2^2\big) &= \mathbb{E}\max_{i\in[n]}\sum_{a=1}^d((X_i)_a)^2 \leq \sum_{a=1}^d \mathbb{E}\max_{i\in[n]}\big((X_i)_a\big)^2 \\
&\leq \sum_{a=1}^d \frac{1}{t}\log\big(\mathbb{E}\big(e^{t\max_{i\in[n]}((X_i)_a)^2}\big)\big) \leq \sum_{a=1}^d \frac{1}{t}\log\big(n\mathbb{E}\big(e^{t((X_i)_a)^2}\big)\big),
\end{aligned}$$

for all $t > 0$. We can then bound this by $\sum_{a=1}^d \frac{1}{t}\log(ne^{t(6\sqrt{2e}\sigma_a)^2})$ for $t < 1/(6\sqrt{2e}\sigma_a)^2$, yielding:

$$\mathbb{E}_{\mathcal{D}_n}\big(\max_{i\in[n]}\|X_i\|_2^2\big) \leq 72e(1+\log(n))\sum_{a=1}^d \sigma_a^2.$$

The same proof technique applies to $\mathbb{E}_{\mathcal{D}_n}\left(\max_{i\in[n]}\|X_i\|_\infty^2\right)$, yielding the desired result.

Finally, we consider the subgaussianity of $1+\sqrt{\|X\|^*}$. Note that the sum of two subgaussian variables is subgaussian. Using Vershynin (2018, Proposition 2.5.2 (ii)), for two real random variables Z and $\tilde{Z}$ with variance proxies σ^2 and $\tilde{\sigma}^2$ respectively, we have that $Z+\tilde{Z}$ is subgaussian with variance proxy $(\sigma+\tilde{\sigma})^2$. Additionally, the absolute value of a subgaussian variable is also subgaussian with the same variance proxy (Vershynin, 2018, Proposition 2.5.2).

For $\|\cdot\|=\|\cdot\|_2$, we have $1+\sqrt{\|X\|_2}\le 1+\sum_{a=1}^d |X_a|$. Since 1 and X_a are subgaussian variables, this yields the desired result.

For $\|\cdot\|_\infty$, for all $t>0$,

$$\begin{aligned}\mathbb{P}(\|X\|_\infty \ge \sqrt{2\sigma^2\log(2d)}+t) &\le 2de^{-\frac{(\sqrt{2\sigma^2\log(2d)}+t)^2}{2\sigma^2}}\\ &\le 2e^{-\frac{t^2}{2\sigma^2}-\frac{t\sqrt{\log(2d)}}{\sqrt{2\sigma^2}}} \le 2e^{-\frac{t^2}{2\sigma^2}}.\end{aligned}$$

Thus, $\|X\|_\infty-\sqrt{2\sigma^2\log(2d)}$ is subgaussian with variance proxy σ^2. Therefore, $\|X\|_\infty$ is subgaussian with variance proxy bounded by $\sigma^2(1+\sqrt{\log(2d)})^2$. Then, $1+\sqrt{\|X\|_\infty}$ is subgaussian because it is less than $2+\|X\|_\infty$, which is subgaussian (Vershynin, 2018, Proposition 2.5.2) with a variance proxy bounded by that of $2+\|X\|_\infty$, yielding the desired result. ■

A.5 Lemmas Needed for Section 4.2 and their Proofs

Here we provide lemmas necessary for the proof of Theorem 16 and the analysis of its distribution-dependent terms.

A.5.1 Lemma 19 and its Proof

Lemma 19 relates the Gaussian complexity to useful quantities to bound the expected risk.

Lemma 19 *(Use of Gaussian Complexity) Let $D>0$ and the data set $\mathcal{D}_n=(x_i,y_i)_{i\in[n]}$ consists of i.i.d. samples of the random variable $(X,Y)\in\mathcal{X}\times\mathcal{Y}$. Assume that the loss ℓ is L-Lipschitz in its second (bounded) argument, i.e., $\forall y\in\mathcal{Y}, a\in\{f(x)\mid x\in\mathcal{X},\ f\in\mathcal{F}_\infty,\ \Omega(f)\le D\}, a\to\ell(y,a)$ is L-Lipschitz. Then, we have*

$$\mathbb{E}_{\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\le D}\widehat{\mathcal{R}}(f)-\mathcal{R}(f)+\sup_{f\in\mathcal{F}_\infty,\Omega(f)\le D}\mathcal{R}(f)-\widehat{\mathcal{R}}(f)\right)\le 6DL\left(\frac{1}{\sqrt{n}}+G_n\right).$$

Proof [Lemma 19] By Bach (2024, Proposition 4.2), we have

$$\begin{aligned}&\mathbb{E}_{\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\le D}\widehat{\mathcal{R}}(f)-\mathcal{R}(f)+\sup_{f\in\mathcal{F}_\infty,\Omega(f)\le D}\mathcal{R}(f)-\widehat{\mathcal{R}}(f)\right)\\ &\le 4\mathbb{E}_{\tilde{\varepsilon},\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\le D}\frac{1}{n}\sum_{i=1}^n\varepsilon_i\ell(y_i,f(x_i))\right),\end{aligned}$$

where $\tilde{\varepsilon}$ consists of i.i.d. Rademacher variables.

Next, applying the contraction principle from Bach (2024, Proposition 4.3), we get

$$\mathbb{E}_{\tilde{\varepsilon},\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D}\frac{1}{n}\sum_{i=1}^n\varepsilon_i\ell(y_i,f(x_i))\right)\leq\mathbb{E}_{\tilde{\varepsilon},\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D}\frac{1}{n}\sum_{i=1}^n\tilde{\varepsilon}_i f(x_i)\right).$$

Then, using Wainwright (2019, Exercise 5.5), we have

$$\mathbb{E}_{\tilde{\varepsilon},\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D}\frac{1}{n}\sum_{i=1}^n\tilde{\varepsilon}_i f(x_i)\right)\leq\sqrt{\frac{\pi}{2}}\mathbb{E}_{\varepsilon,\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D}\frac{1}{n}\sum_{i=1}^n\varepsilon_i f(x_i)\right),$$

where $\varepsilon\sim\mathcal{N}(0,I_d)$.

Finally, by applying Lemma 11 and combining all these results, we obtain the desired inequality

$$\mathbb{E}_{\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D}\widehat{\mathcal{R}}(f)-\mathcal{R}(f)+\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D}\mathcal{R}(f)-\widehat{\mathcal{R}}(f)\right)\leq 6DL\left(\frac{1}{\sqrt{n}}+G_n\right).$$

■

A.5.2 Lemma 20 and its Proof

Lemma 20 describes a useful property on the expectation of the hyperbolic cosine of a subgaussian random variable.

Lemma 20 *(Technical Lemma on Subgaussian Random Variables) Let Z be a real-valued random variable (not necessarily centred) that is subgaussian (see Definition 15.) Then, for all $\lambda\in\mathbb{R}$,*

$$\mathbb{E}\left(\cosh(\lambda Z)\right)\leq e^{(6\sqrt{2e})^2\sigma^2\lambda^2}.$$

Proof [Lemma 20] An equivalent definition of subgaussianity is that for all $\lambda\in\mathbb{R}$, if $6\sqrt{2e}\sigma|\lambda|\leq 1$, then $\mathbb{E}(e^{\lambda^2Z^2})\leq e^{(6\sqrt{2e})^2\sigma^2\lambda^2}$, see Vershynin (2018, Proposition 2.5.2).

First, in the case $|\lambda|\leq\frac{1}{6\sqrt{2e}\sigma}$. Using the inequality $e^x\leq x+e^{x^2}$ for all $x\in\mathbb{R}$, we get

$$\mathbb{E}\left(\cosh(\lambda Z)\right)\leq\mathbb{E}\left(\frac{\lambda Z+e^{\lambda^2Z^2}-\lambda Z+e^{\lambda^2Z^2}}{2}\right)=\mathbb{E}\left(e^{\lambda^2Z^2}\right)\leq e^{(6\sqrt{2e})^2\sigma^2\lambda^2}.$$

Next, consider the case $|\lambda|\geq\frac{1}{6\sqrt{2e}\sigma}$. We can bound the expectation as follows

$$\begin{aligned}\mathbb{E}\left(\cosh(\lambda Z)\right)&\leq\mathbb{E}\left(e^{|\lambda Z|}\right)=\mathbb{E}\left(e^{6\sqrt{2e}\sigma|\lambda|\frac{|Z|}{6\sqrt{2e}\sigma}}\right)\leq\mathbb{E}\left(e^{(6\sqrt{2e})^2\sigma^2\lambda^2/2+\frac{Z^2}{2(6\sqrt{2e})^2\sigma^2}}\right)\\&\leq e^{(6\sqrt{2e})^2\sigma^2\lambda^2/2}e^{1/2}\leq e^{(6\sqrt{2e})^2\sigma^2\lambda^2},\end{aligned}$$

where we use the fact that $(6\sqrt{2e})^2\sigma^2\lambda^2\geq 1$ to justify the final inequality.

Thus, in both cases, we have shown that $\mathbb{E}\left(\cosh(\lambda Z)\right)\leq e^{(6\sqrt{2e})^2\sigma^2\lambda^2}$, proving the lemma.

■

A.5.3 Lemma 21 and its Proof

Lemma 21 is an application of McDiarmid's inequality (a specific version by Meir and Zhang (2003) for subgaussian random variables) to our learning problem.

Lemma 21 *(Use of McDiarmid's Inequality) Let $D > 0$ and $\delta \in (0,1)$. Assume that $1 + \sqrt{\|X\|^*}$ is subgaussian with variance proxy σ^2 and that the loss ℓ is L-Lipschitz in its second (bounded) argument, i.e., $\forall y \in \mathcal{Y}, a \in \{f(x) \mid x \in \mathcal{X},\ f \in \mathcal{F}_\infty,\ \Omega(f) \leq D\}, a \to \ell(y,a)$ is L-Lipschitz. Then, with probability greater than $1-\delta$,*

$$\begin{aligned}
&\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D} \widehat{\mathcal{R}}(f) - \mathcal{R}(f) + \sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D} \mathcal{R}(f) - \widehat{\mathcal{R}}(f) \\
&\qquad \leq \mathbb{E}_{\mathcal{D}_n}\left(\sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D} \widehat{\mathcal{R}}(f) - \mathcal{R}(f) + \sup_{f\in\mathcal{F}_\infty,\Omega(f)\leq D} \mathcal{R}(f) - \widehat{\mathcal{R}}(f) \right) \\
&\qquad\quad + \frac{48\sqrt{2e}LD\sigma}{\sqrt{n}} \sqrt{\log \frac{1}{\delta}}.
\end{aligned}$$

Proof [Lemma 21] We use a specific version of McDiarmid's inequality (Meir and Zhang, 2003, Theorem 3). First, we show that the conditions for applying the theorem are met. Let $\tilde{H} := \{h : (x,y) \in \mathcal{X}\times\mathcal{Y} \to \ell(y,f(x)) - \ell(y,\tilde{f}(x)) \mid \Omega(f) \leq D, \Omega(\tilde{f}) \leq D\}$. For any $\lambda > 0$, we have

$$\begin{aligned}
&\mathbb{E}_{X,Y}\left(\sup_{h,\tilde{h}\in\tilde{H}} \cosh(2\lambda(h(X,Y) - \tilde{h}(X,Y))) \right) \\
&\qquad = \mathbb{E}_{X,Y}\left(\sup_{f,\Omega(f)\leq D,\tilde{f},\Omega(\tilde{f})\leq D} \cosh(2\lambda(\ell(Y,f(X)) - \ell(Y,\tilde{f}(X)))) \right) \\
&\qquad \leq \mathbb{E}_{X,Y}\left(\sup_{f,\Omega(f)\leq D,\tilde{f},\Omega(\tilde{f})\leq D} \cosh(2\lambda L|f(X) - \tilde{f}(X)|) \right) \\
&\qquad \leq \mathbb{E}_{X,Y}\left(\sup_{f,\Omega(f)\leq D,\tilde{f},\Omega(\tilde{f})\leq D} \cosh(4\lambda LD(1+\sqrt{\|X\|^*})) \right) \\
&\qquad = \mathbb{E}_{X,Y}\left(\cosh(4\lambda LD(1+\sqrt{\|X\|^*})) \right) \leq e^{(48\sqrt{e})^2 L^2 D^2 \sigma^2 \lambda^2/2},
\end{aligned}$$

where the last inequality follows from Lemma 20. Hence, the condition is verified with $M = 48\sqrt{e}LD\sigma$ and applying Meir and Zhang (2003, Theorem 3) yields the desired result. ■

Appendix B. Numerical Experiments

In this section, we detail the parameters and methodology used in the different experiments. The code needed to run the experiments can be found at `https://github.com/BertilleFollain/BKerNN`.

B.1 Experiment 1: Optimisation procedure, Importance of Positive Homogeneous Kernel

Each method was tuned using 5-fold cross-validation with grid search, using negative mean squared error as the scoring metric. The training was set for 20 iterations and the step-size parameter (γ) was set to 500, with backtracking enabled. Regularisation parameter candidates were $\lambda = \{0.05, 0.1, 0.5, 1, 1.5\} \times 2 \max_{i\in[n]} \|x_i\|_2/n$. Once the regularisation parameters had been selected, we trained from scratch for 200 iterations, with the other parameters kept as before.

B.2 Experiments 2 & 3: Influence of Parameters (Number of Particles m, Regularisation Parameter λ, and Type of Regularisation)

For Experiment 2, in the first subplot, we set the step-size parameter γ to 500 and the number of iterations to 50. The regularisation type was set to Ω_{basic} and the regularisation parameter to $\lambda = 0.02$. The tested values of m were 1, 3, 5, 7, 10, 15, 20, 30, 40, and 50.

In the second subplot, we varied the regularisation parameter λ in 0.0005, 0.001, 0.005, 0.01, 0.02, 0.05, 0.1, 0.3, and 0.5, while keeping the number of particles fixed at $m = 10$.

In Experiment 3, the BKERNN model was instantiated with a fixed number of particles $m = 20$, step-size parameter $\gamma = 500$, and number of iterations 25. The regularisation parameter λ was set as $2 \max_{i\in n} \|x_i\|_2/n$.

B.3 Experiment 4: Comparison to Neural Network on 1D Examples, Influence of Number of Particles/Width of Hidden Layer m

In Experiment 4, we investigated the performance of two learning methods, BKERNN and RELUNN, on three different 1D functions. The training set always consists of 128 samples, with x sampled uniformly between -1 and 1, while the target function/test set without noise consists of 1024 equally spread out points. The response was then generated as follows. For the first function, $y = \sin(2\pi x) + \text{noise}$, for the second $y = \text{sign}(\sin(2\pi x)) + \text{noise}$, for the third $y = 4|x + 1 - 0.25 - \lfloor x + 1 - 0.25 \rfloor - 0.5| - 1 + \text{noise}$, where the noise is always normal, centred and with standard deviation equal to 0.2. For BKERNN, the regularisation parameter λ was selected from [0.005, 0.01, 0.02, 0.05] using 5-fold cross-validation and the negative mean squared error score. RELUNN was trained using a batch size of 16, a number of iterations equal to 400,000 and a step-size of 0.005.

B.4 Experiment 5: Prediction Score and Feature Learning Score Against Growing Dimension and Sample Size, a Comparison of BKERNN with Kernel Ridge Regression and a ReLU Neural Network

In Experiment 5, data sets were generated with input data uniformly sampled within the hypercube $[-1, 1]^d$. The feature matrix P was generated from the orthogonal group. For each configuration, training and test sets of sizes n and $n_{\text{test}} = 201$, respectively, were created. Output labels y were computed as $y_i = |\sum_{a=1}^{k} (\sin((P^\top x_i)_a))|$, with $k = 3$ relevant features.

The first two plots fixed the dimension at 15 and varied sample sizes across [10, 20, 50, 100, 150, 200, 300, 400, 500]. The last two plots fixed the sample size at 212 and varied

dimensions across [3, 5, 10, 20, 30, 40, 50]. Each configuration was repeated 10 times with different random seeds.

For BKerNN: λ was set to $2\max_{i\in[n]}(\|x_i\|_2)/n$, the number of particles was $m = 50$, the regularisation type Ω_{feature}, the number of iterations 20, and step-size $\gamma = 500$ with backtracking line search. For BKRR, λ was chosen similarly to BKerNN. For ReLUNN, the number of neurons was set to 50, learning rate to 0.05, batch size to 16, and number of iterations to 1500.

B.5 Experiment 6: Comparison on Real Data Sets Between BKerNN, Kernel Ridge Regression and a ReLU Neural Network

In Experiment 6, BKRR and both versions of BKerNN had regularisation parameter fixed equal to $\max_{i\in[n]}(\|x_i\|_2)/n$, where n is the number of training samples (i.e. 400). Backtracking line search was used for BKerNN and the starting step-size was 500, while the number of iterations was 40. For ReLUNN, the batch size was 16, while the number of iterations was 2500 which corresponds to 100 epochs, and the step-size was set to 0.01.

References

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

Francis Bach. *Learning Theory from First Principles*. MIT Press, 2024. URL `https://www.di.ens.fr/~fbach/ltfp_book.pdf`. to appear.

Francis Bach, Gert Lanckriet, and Michael Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *International Conference on Machine Learning (ICML)*, 2004.

Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1): 1–106, 2012.

Peter Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Amir Beck. *First-Order Methods in Optimization*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, 2017.

Alain Berlinet and Christine Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media, 2011.

Alberto Bietti, Joan Bruna, and Loucas Pillaud-Vivien. On learning gaussian multi-index models with gradient flow, 2023. URL `https://arxiv.org/abs/2310.19793`.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

David Brillinger. A generalized linear model with "Gaussian" regressor variables. In *Selected Works of David Brillinger*, pages 589–606. Springer, 2012.

Lénaïc Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

Lénaïc Chizat and Francis Bach. Gradient descent on infinitely wide neural networks: global convergence and generalization. In *Proceedings of the International Congress of Mathematicians 2022*, pages 5398–5419. EMS Press, 2022.

Arnak Dalalyan, Anatoli Juditsky, and Vladimir Spokoiny. A new algorithm for estimating the effective dimension-reduction subspace. *Journal of Machine Learning Research*, 9(53): 1647–1678, 2008.

Petros Drineas and Michael Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6(72): 2153–2175, 2005.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

Bertille Follain and Francis Bach. Nonparametric linear feature learning in regression through regularisation, 2024. URL `https://arxiv.org/abs/2307.12754`.

Kenji Fukumizu, Francis Bach, and Michael Jordan. Kernel dimension reduction in regression. *The Annals of Statistics*, 37(4):1871–1905, 2009.

Christophe Giraud. *Introduction to High-Dimensional Statistics*. CRC Press, 2014.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12(64):2211–2268, 2011.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 507–520, 2022.

Vladimir Koltchinskii, Karim Lounici, and Alexandre Tsybakov. Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion. *The Annals of Statistics*, 39(5): 2302–2329, 2011.

Gert Lanckriet, Nello Cristianini, Peter Bartlett, Laurent Ghaoui, and Michael Jordan. Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, 5(Jan):27–72, 2004.

Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes*. A Series of Modern Surveys in Mathematics Series. Springer, 1991.

Ker-Chau Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991.

Fanghui Liu, Leello Dadi, and Volkan Cevher. Learning with norm constrained, over-parameterized, two-layer neural networks. *Journal of Machine Learning Research*, 25 (138):1–42, 2024.

Ulrike Luxburg and Olivier Bousquet. Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research*, 5(Jun):669–695, 2004.

Pierre Marion and Raphaël Berthier. Leveraging the two-timescale regime to demonstrate convergence of neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 64996–65029, 2023.

Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Mean-field theory of two-layer neural networks: Dimension-free bounds and kernel limit. In *Conference on Learning Theory (COLT)*, volume 99, pages 2388–2464, 2019.

Ron Meir and Tong Zhang. Generalization error bounds for Bayesian mixture algorithms. *Journal of Machine Learning Research*, 4(Oct):839–860, 2003.

Yuliya Mishura and Georgiy Shevchenko. Gaussian processes. integration with respect to gaussian processes. In *Theory and Statistical Applications of Stochastic Processes*, chapter 3, pages 39–65. John Wiley & Sons, Ltd, 2017.

Atsushi Nitanda and Taiji Suzuki. Stochastic particle gradient descent for infinite ensembles, 2017. URL `https://arxiv.org/abs/1712.05438`.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.

Lorenzo Rosasco, Silvia Villa, Sofia Mosci, Matteo Santoro, and Alessandro Verri. Non-parametric sparsity and regularization. *Journal of Machine Learning Research*, 14(52): 1665–1714, 2013.

Bernhard Schölkopf, Ralf Herbrich, and Alex Smola. A generalized representer theorem. In *Computational Learning Theory (COLT)*, pages 416–426, 2001.

Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, 80(2):725–752, 2020.

Endre Süli and David Mayers. *An Introduction to Numerical Analysis*. Cambridge University Press, 2003.

Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.

Vladimir Vovk. Kernel ridge regression. In *Empirical inference*, pages 105–116. Springer, 2013.

Grace Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, 1990.

Martin Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019.

Christopher Williams and Carl Edward Rasmussen. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Sewall Wright. Correlation and causation. *Journal of Agricultural Research*, 20(3):557–585, 1921.

Yingcun Xia. A multiple-index model and dimension reduction. *Journal of the American Statistical Association*, 103(484):1631–1640, 2008.

Yingcun Xia, Howell Tong, Wai Keung Li, and Li-Xing Zhu. An adaptive estimation of dimension reduction space. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 64(3):363–410, 2002.

Zhuoran Yang, Krishnakumar Balasubramanian, Zhaoran Wang, and Han Liu. Estimating high-dimensional non-gaussian multiple index models via stein's lemma. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Yiming Ying and Colin Campbell. Rademacher Chaos Complexities for Learning the Kernel Problem. *Neural Computation*, 22(11):2858–2886, 11 2010.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.