UNIVERSITY
OF OSLO

Scaling kernel-based learning for big data

Andreas Oslandsbotn

Supervisors

Alexander Cloninger
Nickolas Forsch
Željko Kereta
Aslak Tveito

Department of Informatics

The Faculty of Mathematics and Natural

Sciences

2023

Series of dissertations submitted to the
Faculty of Mathematics and Natural Sciences, University of Oslo
No. 2698

ISSN 1501-7710

Cover: UiO.
Print production: Graphic center, University of Oslo.

Under linden

Sitte på ein stubbe under linden
I den leikande varme augustvinden
Lye til biene sitt summekor
Der dei strevar og samlar inn vinterfor
Kjenne angen av roser og kaprifol
Og varmande strålar frå seinsommarsol
Nyte selskap med erla og raudstrupen vår
Som tok ferien sin her i hagen i år
Han svinsar og sprett ifrå grein til grein
So syng han ei strofe so klar og so rein
Og erla ho trippar og leitar seg føde
Ho er ferdig for i år med barnestell og møde
Så no skal me nyte vår otium me tre
Her i hagen, under linden, i ro og i fred

Haldis Brenne (1920 – 2011)

Abstract

In the face of datasets with ever increasing sizes, development of scalable learning algorithms with high predictive power is at the forefront of machine learning research. The design of learning schemes using a positive semi-definite function at the core (a kernel function), is motivated by a solid theoretical foundation. These schemes have shown great success on moderately sized datasets. However, kernel-based methods suffers from scalability issues, due to their inherent large memory requirements and computational costs. In this thesis we address this issue by developing kernel-based learning schemes that are compatible with efficient computational models such as streaming, paralellization and distribution. We support the proposed algorithms with theoretical and numerical results.

Samandrag

I møte med store datamengder, er utvikling av skalerbare læringsalgoritmar med gode generaliseringseigenskapar eit viktig forskingsområde i maskinlæring. Konstruksjon av læringsmetoder som utnyttar en positiv semi-definit funksjon (ein kernelfunksjon), er ein strategi som har vist stor suksess ved moderate datamengder og er støtta av eit solid teoretisk grunnlag. Desverre er kernelbaserte metodar avgrensa frå bruk i møte med store datamengder, på grunn av betydelege minne og kalkulasjonsbehov. I denne doktorgraden løyser vi dette problemet ved å utvikle kernelbaserte læringsmetoder som er kompatible med effektive modeller slik som parallellisering, distribuerte system og strømming av data.

Contents

List of Figures

List of Tables

Preface

This thesis is submitted for the degree of *Philosophiae Doctor* at the University of Oslo. The research presented in this thesis was conducted at Simula Research Laboratory from September 2019 until September 2021 and then at the Department of Mathematics, University of California San Diego, between October 2021 and March 2023, before returning to Simula Research Laboratory from April 2023 until June 2023.

The thesis has been conducted under the supervision of Professor Alexander Cloninger at the Department of Mathematics, University of California San Diego, Research Engineer Nickolas Forsch at the Department of Computational Physiology, Simula Research Laboratory, and Željko Kereta, Research Fellow at the Department of Computer Science, University College London. In the supervisor team has also been Professor Aslak Tveito at the Department of Informatics, University of Oslo.

This document comprises an introduction and four original manuscripts; two first-author papers and two co-first-author papers. The introduction provides an overview of the field and clarifies the contribution of the individual manuscripts.

Acknowledgements

I would like to express my gratitude to my supervisor Prof. Alexander Cloninger for following me on this journey; your mathematical intuition and openness to discussions have been greatly appreciated. Furthermore, I would like to thank my supervisor Nickolas Forsch for all his support in progressing this thesis and for the valuable insights he has provided on cardiac physiology. Finally, I would like to thank my supervisor Željko Kereta for the discussions, the support, and the eye for mathematical detail.

I would also like to thank the other researchers I have met during my work on this thesis. In particular, I would like to thank Prof. Yoav Freund for his creative ideas and Robi Bhattacharjee, Zhengchao Wan, Sawyer Robertson, and Stefano Vigogna for their collaboration and discussions.

My gratitude also goes to the other PhD students I have met at Simula. A particular warm thanks to the PhD students at the Suurph program and all the great experiences we have shared. I will also thank Kimberly McCabe for her commitment to me and the other suurphers, helping out with everything around the PhD.

Finally, I would like to thank Maria and my family, who have been there to support me during this long endeavor.

Introduction

Recent advances in computational power, memory capabilities, sensor technology, and the internet have allowed the collection and storage of increasingly large data sets. The size of these data sets is not only due to a massive growth in the number of collected samples but also due to a substantial increase in the number of collected attributes (observable quantities) associated with each sample.

In machine learning and data analysis, we are interested in identifying patterns in data sets to gain insights that can be used in real-world applications. The patterns of interest found in data can be highly non-linear. Therefore, these patterns are often infeasible to model with existing elementary functions containing few tunable parameters, such as linear functions, polynomials, Gaussians, etc. Furthermore, with multiple attributes, the data can be difficult to visualize, preventing insights that could otherwise guide the choice of the model function. Consequently, parametric modeling with elementary functions is limited in the face of non-linear patterns.

The limitations of parametric learning have motivated the development of non-parametric learning schemes that do not rely on strong modeling assumptions and pre-defined knowledge about the data. Notable examples are kernel-based methods, decision trees, and neural networks.

In this thesis, we focus on *kernel-based* methods, meaning all methods that map non-linear input data to a high-dimensional feature space using the so-called kernel function. In doing so, non-linear data dependencies are approximately linearized, allowing the usage of fast and efficient linear models. Kernel-based methods came to prominence at the end of the 1990s [105] with the introduction of the support vector machine (SVM) [32] for non-linear classification, kernel principal component analysis (kernel PCA) [102, 103] for non-linear PCA and kernel ridge regression (KRR) [99, 104] for non-linear regression.

Another common strategy to capture non-linear patterns, also relying on the kernel function, are spectral embedding methods based on nearest neighbor graphs such as ISOMAP [115], locally linear embedding (LLE) [93], Laplacian eigenmaps (LE) [14, 16], and diffusion maps [31].

The popularity of kernel-based methods stems from their solid and studied theoretical foundation [14, 31, 92, 101, 105]. However, kernel-based learning methods generally have large memory and computational requirements due to their reliance on a kernel matrix that scales with the number of training samples.

Consequently, in real-world applications, these methods have largely fallen out of favor in the machine learning community with the rise of alternative methods such as multi-layer neural networks.

The motivating principle of this thesis is that reducing computational costs and memory requirements alone, does not unlock the full potential for scalability of learning methods. Rather, the use of modern computational models, such as parallelization, distribution, and learning from streaming data, is also a critical ingredient. In this thesis, we develop techniques and algorithms for the purpose of scaling up kernel-based learning schemes for big data applications. Our strategy is two-fold. On one hand, by utilizing ideas and concepts such as tailored sub-sampling, boosting, multi-resolution, sparsity, and iterative solvers, we develop learning schemes with low memory requirements and low computational cost. At the same time, the learning methods developed in this work are designed to be compatible with modern computational models, such as those mentioned above, that are essential for large scale applications. In summary, we aim to develop learning schemes that satisfy the following aspects:

1. *Single-pass* - learning schemes that can learn from seeing each sample only once before discarding it.

2. *Distributed* - learning schemes that can be divided into independent modules that can learn independently or with minimal communication.

3. *Minimize in-memory data* - learning schemes with independent modules that only require access to a fraction of the data.

We consider a setting where data is embedded in a high-dimensional ambient space, the Euclidean space $\mathbb{R}^D$. Furthermore, we assume that the data comes from a distribution supported on, or concentrated around, a lower-dimensional set $\mathcal{X} \in \mathbb{R}^D$. We assume that $\mathcal{X}$ is a point cloud, a notion we define in more detail later. In particular, we consider point clouds where the intrinsic structure can be highly non-linear and vary in dimensionality.

We develop scalable regression algorithms that can learn highly non-linear functions and develop techniques that reduce the impact of the *curse of dimensionality* [122]. Furthermore, we develop scalable algorithms that can learn the intrinsic structure of data for the purpose of dimensionality reduction and quantifying similarity between samples in a non-linear space.

The algorithms we develop are verified numerically and supported by theoretical analysis.

Outline In Section 1 we present an overview of relevant concepts related to non-parametric learning in a supervised setting. We then discuss the curse of dimensionality and associated challenges in Section 2. In Section 3 we introduce the concept of point clouds and discuss challenges and opportunities related to uncovering their intrinsic structure. Section 4 then discusses methods for uncovering the structure in point clouds for purposes such as dimensionality reduction. Section 5 reviews several strategies for scaling learning algorithms to

big data applications. In Section 6, we present the contributions of the original manuscripts underlying the research performed in this thesis. Finally, Section 7 concludes with a summary of the findings in the thesis and discusses further work.

Notation We denote matrices with upper case and vectors with lower case. For a matrix $A \in \mathbb{R}^{n\times n}$ we let A_{ij} be the i,j-th entries. For a vector $a \in \mathbb{R}^n$, we let a_i denote its i-th entry. For $\Gamma, \Gamma' \subseteq \{1, \dots n\}$ we let $A(\Gamma, \Gamma')$ mean the sub-matrix constructed from the indices contained in Γ, Γ'. We let $A^\top$ denote the transpose of A and $A^\dagger$ the Moore-Penrose pseudoinverse. For a vector $v \in \mathbb{R}^D$ we let $\|v\|_p$ denote the p-norm. For a function $f : \mathcal{X} \to \mathbb{R}$ and a set of samples $\{x_i\}_{i=1}^n$ we take $f([x_n])$ to mean $f([x_n]) = (f(x_1), f(x_2), \dots, f(x_n))^\top$.

For a distribution ρ we mean by $X \sim \rho$ that X is a random variable distributed according to ρ. We let $\mathbb{E}_{X\sim\rho}[X]$ be the expectancy of X. For random variables $(X, Y) \sim \rho$ we let $\mathbb{E}_{(X,Y)\sim\rho}[Y|X = x]$ be the expectancy of Y conditioned on X. By $\mathcal{N}(\mu, \sigma)$, we refer to the normal distribution with mean μ and covariance σ. We mean by $\mathrm{Uni}(\Omega)$ the uniform distribution over a set $\Omega \subset \mathbb{R}^D$.

We let $L^2(\mathcal{X}, \rho_\mathcal{X})$ be the space of square integrable functions with norm $\|f\|_\rho^2 = \int_\mathcal{X} |f(x)|^2 d\rho_\mathcal{X}$. For a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ we let $\mathcal{H}_k$ denote the reproducing kernel Hilbert space induced by k. The inner product in $\mathcal{H}_k$ is defined as $\langle f, g\rangle_k = \sum_{ij} \alpha_i, \beta_j k(x_i, x_j)$, for $g, f \in \mathcal{H}_k$. The associated norm is $\|\cdot\|^2 = \langle \cdot, \cdot \rangle_k$. We denote by (M, d) a metric space with distance metric d.

1 Supervised nonlinear function learning

Consider the problem of learning the relationship between some response $y \in \mathbb{R}$ and an input $x \in \mathcal{X} \subseteq \mathbb{R}^D$ for the purpose of predicting the response y given a new input sample $x \in \mathcal{X}$. This problem is often addressed in a supervised setting and is encountered in numerous applications. For example in bio-medicine, knowing the relationship between ionic membrane currents x and the action potential in cardiac cells y allows predicting the effects of specific treatments during drug development [57, 118].

We can formulate this problem in the framework of statistical learning theory [34, 121]. In this setting, the input-response pair (x, y) is interpreted as the realization of a random variable (X, Y) sampled from a probability distribution $\rho = \rho(y|x)\rho_\mathcal{X}$. The probability distribution ρ is assumed to be unknown and can only be accessed through a finite set of training samples $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^n$.

The relationship between input and response is described by the conditional distribution $\rho(y|x)$. However, finding an exact description of this distribution is often infeasible. In regression, the goal is instead to learn a target function $f : \mathcal{X} \to \mathbb{R}$, using the available training samples $\mathcal{D}_n$, that gives a suitable approximation of $\rho(y|x)$.

To learn the target function, it is necessary to define a loss that quantifies estimation quality and guides the learning process. The L_2 error is a popular choice because it is mathematically easy to work with and often gives optimization problems that are computationally cheaper to evaluate than, say, other L_p loss

functions [49]. Function learning can then be formulated as the minimization of the expected L_2 risk

$$\mathcal{E}(f) = \mathbb{E}_{(X,Y)\sim\rho}[(f(X) - Y)^2], \tag{1.1}$$

for $f \in L^2(\mathcal{X}, \rho_{\mathcal{X}})$, where $L^2(\mathcal{X}, \rho_{\mathcal{X}})$ is the space of square integrable functions. For the remainder of the text, we take $\mathbb{E}$ to mean $\mathbb{E}_{(X,Y)\sim\rho}$ unless otherwise stated.

Remark 1 (The regression function) *The minimizer of the expected L_2 risk can be shown to be the conditional mean $f_\rho(x) = \mathbb{E}_{Y\sim\rho(y|x)}[Y|X = x]$, called the regression function. However, since ρ is known only through a finite sample, the regression function f_ρ cannot be calculated explicitly. Instead, it can be estimated using the training samples in $\mathcal{D}_n$.*

The classical way to estimate f_ρ is by assuming it can be modeled with an explicit function class of elementary functions, such as polynomials or Gaussians. This approach is known as parametric regression, with linear and polynomial regression as typical examples. The problem with parametric regression is that prior knowledge of the shape and characteristics of f_ρ, such as linearity vs. non-linearity, differentiability, continuity, etc., is rarely known. This is especially problematic in multi-feature settings where visualization of the data is impractical or infeasible, which prevents further insights. Since choosing the wrong model can cause significant errors, parametric learning is limited to simpler learning tasks.

1.1 Non-parametric learning in a hypothesis space

Non-parametric regression, in contrast to parametric regression, does not require pre-defined knowledge of the target function; see Györfi et al. [49]. Nevertheless, learning in non-parametric settings still requires a model that, in some way, can be tuned to the data. The simplest approach is to use estimators that rely on local averages, such as kNN and kernel smoothers. A well-established alternative is to introduce a hypothesis space $\mathcal{H} \subseteq L^2(\mathcal{X}, \rho_{\mathcal{X}})$, in which estimators of f_ρ are pursued. It is important to note that the regression function f_ρ need not be contained in $\mathcal{H}$.

The hypothesis space $\mathcal{H}$ provides the necessary structure to define a model that can be fitted to the data. Examples are the spaces of piece-wise polynomials, splines, or reproducing kernel Hilbert spaces (RKHS). The parameters p defining these hypothesis spaces, such as the degree of the polynomial or the bandwidth of the kernel inducing the RKHS, are normally referred to as hyper-parameters and can be tuned to the data to improve the model. In Section 1.2, we discuss the selection of these parameters in more detail.

When searching for an optimal estimator $\widehat{f}_{n,\lambda}$ in a hypothesis space $\mathcal{H}$, the minimizer of the expected L_2 risk $\mathbb{E}[(f(X) - Y)^2]$ is normally estimated using a penalized version of the empirical risk minimizer

$$\widehat{f}_{n,\lambda} = \operatorname*{argmin}_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 + J_\lambda(f). \tag{1.2}$$

Here $J_\lambda(f)$ is a penalty on the complexity of f, and λ is a hyper-parameter that governs the magnitude of the penalty. We can think of λ as restricting the available hypothesis space to less complex functions, as illustrated in Figure 1.1.

To solve the minimization problem in Eq. (1.2), it is necessary to make explicit choices on the hypothesis space $\mathcal{H}$. In Section 1.4, we discuss a specific choice relevant to the work in this thesis, namely that of a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$.

1.2 Choosing the hyper-parameters

When estimating a function from finite samples, it is important to consider the bias-variance trade-off [51] and the danger of overfitting to the training samples. The empirical mean of the L_2-loss, namely

$$\mathcal{E}_n(f) = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2,$$

only penalizes the discrepancy from the available data $\mathcal{D}_n$. Consequently, the minimization of $\mathcal{E}_n(f)$ prefers an estimator that interpolates the training samples, but offers no guarantees to predict the response from new samples. To avoid overfitting, hyper-parameters p are introduced to restrict the complexity of the estimator $\widehat{f}_{n,p}$. These hyper-parameters can, for example, be a penalty on the complexity of the estimator, such as λ in the optimization problem defined in Eq. (1.2). Alternatively, they can be parameters restricting the size of the hypothesis space directly, such as the bandwidth of a Gaussian kernel inducing an RKHS. In Györfi et al. [49], the learning goal is stated as finding the hyper-parameters p that minimize the expected L_2 risk

$$\min_p \mathbb{E}_{(X,Y)\sim\rho}[\widehat{f}_{n,p}(X) - Y|\mathcal{D}_n]$$

conditioned on the training data on which $\widehat{f}_{n,p}$ has been trained.

To select the hyper-parameters, it is necessary to evaluate the estimator $\widehat{f}_{n,p}$ on new samples. Since the only available samples are $\mathcal{D}_n$, this is typically done by splitting $\mathcal{D}_n$ into a training set, a validation set, and a test set. The validation set is then used to tune the hyper-parameters, and the test set to evaluate the final generalization performance. Tuning the hyper-parameters based on a single validation set is vulnerable to biased estimation [49]. Techniques such as leave-one-out [25] and k-fold cross-validation [6] are often used to mitigate this issue. The central idea is to split the training set into equalized batches. Then in each iteration, one batch of data is held out during training to evaluate the estimator's prediction capabilities. The mean of the loss over all batches is the final score of a given hyper-parameter. This procedure is repeated for several possible hyper-parameter choices, and the one with the best score is selected.

A major limitation of cross-validation techniques is the high computational cost of repeating the training process over several batches. Furthermore, cross-validation requires the data to be available in memory, but in a streaming setting, this criterion is not satisfied. In the first paper of this thesis, we address this issue in the context of finding a minimizer in an RKHS $\mathcal{H}_k$ induced by a kernel function k_σ. The bandwidth σ of the kernel is a hyper-parameter that must be determined, with implications for the generalization properties of the estimator.

Our strategy is a trade-off between optimizing the hyper-parameter on the one hand and computational cost and compatibility with streaming on the other.

1.3 A note on min-max rates and excess risk

Although we do not analyze optimal estimation rates in this thesis, for completeness and later reference, we introduce a short discussion on min-max rates and excess risk studied in learning theory. The section is based primarily on the discussions in Cucker and Smale [34] and Györfi et al. [49].

When estimating the regression function f_ρ that minimizes the expected L_2 risk using a finite sample $\mathcal{D}_n$ it is of interest to know to what degree the estimator $\widehat{f}_{n,p}$ approximates f_ρ as $n \to \infty$. The study of these rates is done using the min-max approach; see Györfi et al. [49], which seeks lower bounds for the fastest convergence of

$$\inf_{\widehat{f}_n} \sup_{\rho \in \mathcal{C}} \mathbb{E}_{(X,Y)\sim\rho}[(\widehat{f}_n(X) - f_\rho(X))^2]. \tag{1.3}$$

Here the supremum is taken over some class of distributions $\mathcal{C}$ of the random variables (X, Y), and the infimum is taken over all measurable estimators $\widehat{f}_n$ defined on the data $\mathcal{D}_n$ [49]. In other words, one finds a lower bound on the largest excess risk in $L^2(\mathcal{X}, \rho_\mathcal{X})$ over some class of distributions $\mathcal{C}$.

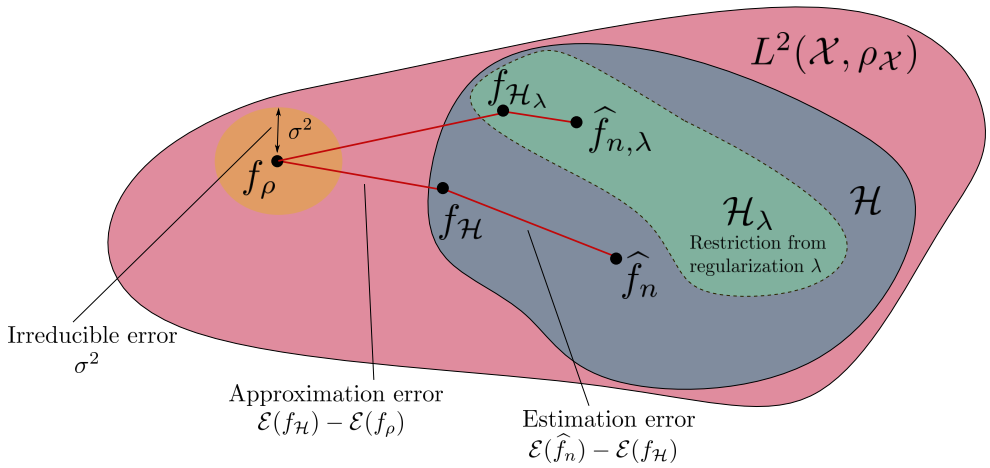


Figure 1.1: Illustration of the hypothesis space $\mathcal{H} \subseteq L^2(\mathcal{X}, \rho_\mathcal{X})$, and the corresponding approximation error $\mathcal{E}(f_\mathcal{H}) - \mathcal{E}(f_\rho)$ and estimation error $\mathcal{E}(\widehat{f}_n) - \mathcal{E}(f_\mathcal{H})$. It is assumed that the noise is additive such that $Y = f_\rho(X) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \sigma)$ and f_ρ is the regression function. $\mathcal{E}(f_\rho) = \sigma^2$ is an irreducible error that can not be improved. The regularization penalizes the complexity of the functions such that hypothesis space is restricted to a subset $\mathcal{H}_\lambda \subseteq \mathcal{H}$. Here $f_{\mathcal{H}_\lambda} = \operatorname{argmin}_{f\in\mathcal{H}_\lambda} \mathcal{E}(f)$. The figure illustrates how the approximation error in $\mathcal{H}_\lambda$ is larger than in $\mathcal{H}$, while the estimation error is smaller. The goal is to find λ such that the overall error $\mathcal{E}(f)$ in Eq. (1.5) is minimized.

In general, the regression function f_ρ is not contained in the hypothesis space $\mathcal{H}$, and the best we can do is $f_\mathcal{H} = \inf_{f\in\mathcal{H}} \mathcal{E}(f)$. This has the consequence of introducing an error $\mathcal{E}(f_\mathcal{H}) - \mathcal{E}(f_\rho)$, known as the approximation error, that can not be reduced by the estimator. We illustrate this situation in Figure 1.1.

Since the approximation error does not depend on the training samples, the min-max rates for an estimator $\widehat{f}_{n,p} \in \mathcal{H}$, are usually studied in terms of the excess risk in $\mathcal{H}$ rather than the excess risk in $L^2(\mathcal{X}, \rho_{\mathcal{X}})$ from Eq. (1.3); See e.g. [24, 34, 95]. We define the excess risk in $\mathcal{H}$ as

$$\mathcal{R}_{\mathcal{H}}(\widehat{f}_{n,p}) = \mathbb{E}[(\widehat{f}_{n,p}(X) - Y)^2] - \inf_{f \in \mathcal{H}} \mathbb{E}[(f(X) - Y)^2]. \tag{1.4}$$

Namely, the error introduced by our finite data estimator $\widehat{f}_{n,p}$ in excess of the error for the best estimator in the hypothesis space. In the literature, it is also common to refer to the excess risk in $\mathcal{H}$ as the estimation or sample error.

We can relate the approximation error and the estimation error to $\mathcal{E}(f)$ through the decomposition

$$\mathcal{E}(f) = \underbrace{\mathcal{E}(\widehat{f}_{n,\lambda}) - \mathcal{E}(f_{\mathcal{H}})}_{\text{Estimation error}} + \underbrace{\mathcal{E}(f_{\mathcal{H}}) - \mathcal{E}(f_\rho)}_{\text{Approximation error}} + \underbrace{\mathcal{E}(f_\rho)}_{\text{Irreducible error}} \tag{1.5}$$

Here, $\mathcal{E}(f_\rho)$ is normally referred to as the irreducible error. Under the assumption that the noise is additive $Y = f_\rho(X) + \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, \sigma)$, we have that $\mathcal{E}(f_\rho) = \sigma^2$. The decomposition is illustrated in Figure 1.1.

We note that the approximation error is normally thought of as a model bias introduced by the choice of hypothesis space. Meanwhile, the estimation error is a variance term arising due to finite training samples. For a fixed sample size n, reducing the size of $\mathcal{H}$ typically increases the approximation error but reduces the estimation error and vice versa. When tuning the hyper-parameters to avoid overfitting, as discussed in section 1.2, it is the trade-off between these two errors we consider. Figure 1.1 illustrates this when the size of the hypothesis space is controlled by the regularization parameter λ. Note that the size of $\mathcal{H}$ can also be controlled by other hyper-parameters, such as the bandwidth of the Gaussian kernel.

1.4 Kernel methods

Kernel methods are a category of non-linear learning algorithms that rely on a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ to create an implicit non-linear embedding of a point cloud $\mathcal{X}$ into a function space equipped with an inner product. The resulting function space is called a reproducing kernel Hilbert space (RKHS), denoted $\mathcal{H}_k$, and allows the use of linear learning schemes.

Definition 2 (Kernel function) *A kernel function is a symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that satisfies the following property*

$$\sum_{i,j} c_i c_j k(x_i, x_j) \geq 0 \quad \forall x_i, x_j \in \mathcal{X}, \quad c_i, c_j \in \mathbb{R}.$$

Given a dataset $\mathcal{D}_n = \{x_i\}_{i=1}^n$ the kernel function induces a positive semi-definite (PSD) matrix $K \in \mathbb{R}^{n \times n}$ with entries $K_{ij} = k(x_i, x_j)$, called the kernel matrix.

The RKHS induced by the kernel function k is defined as

$$\mathcal{H}_k = \Big\{ f : \mathcal{X} \to \mathbb{R} | f(\cdot) = \sum_{i=1}^{\infty} \beta_i k(\cdot, x_i), \beta_i \in \mathbb{R}, x_i \in \mathcal{X}, ||f||_k < \infty \Big\},$$

with inner product $\langle f, g \rangle_k = \sum_{ij} \alpha_i \beta_i k(x_i, x_j)$ for $f(\cdot) = \sum_i \alpha_i k(\cdot, x_i) \in \mathcal{H}_k$ and $g(\cdot) = \sum_j \beta_j k(\cdot, x_j) \in \mathcal{H}_k$. The corresponding norm is denoted $\|\cdot\|^2 = \langle \cdot, \cdot \rangle_k$. Furthermore, the associated feature functions $\phi_i : \mathcal{X} \to \mathbb{R}$, $\phi_i(\cdot) \in \mathcal{H}_k$ can be evaluated implicitly via the "kernel trick" [2, 19]

$$\phi_i(x) = k(x, x_i) = \langle \phi_x, \phi_{x_i} \rangle_k. \tag{1.6}$$

The RKHS is a popular hypothesis space, as it can represent large classes of functions. In fact, for special types of kernels and certain conditions on $\mathcal{X}$, the corresponding $\mathcal{H}_k$ is universal, meaning it contains all bounded continuous functions on $\mathcal{X}$ [79]. Consequently, working in an RKHS allows for representing highly non-linear functions. Moreover, these functions can be efficiently evaluated in the original domain $\mathcal{X}$ using the kernel function k.

Learning in RKHS The theory underlying function approximation in RKHS is well established in machine learning [54] and learning theory [92]. Because of this, kernel methods are generally considered to be theoretically better understood than non-linear learning schemes, such as multi-layer neural networks and decision trees.

Optimizing Eq. (1.2) over the expansion coefficients $\{\beta_i\}_{i=1}^{\infty}$ is generally infeasible as $\mathcal{H}_k$ is normally infinitely dimensional. Therefore, learning in the RKHS is normally done by seeking a minimizer of Eq. (1.2) on the form

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x, x_i), \tag{1.7}$$

provided the optimization problem allows this formulation. Namely, a finite linear expansion in the kernel functions evaluated on the training points $\mathcal{D}_n = \{x_i\}_{i=1}^n$. Seeking an estimator on this form corresponds to reducing the hypothesis space to a finite-dimensional subspace $\mathcal{H}_n \subset \mathcal{H}_k$ defined as

$$\mathcal{H}_n = \{ f \in \mathcal{H}_k : f(\cdot) = \sum_{x_i \in \mathcal{D}_n} \alpha_i k(\cdot, x_i), \alpha_i \in \mathbb{R} \}. \tag{1.8}$$

In many settings, it can be shown that the optimal estimator of Eq. (1.2) is contained in $\mathcal{H}_n$. For example, Schölkopf et al. [101] shows that this is the case for a certain class of loss functions L, when $J_{\lambda,n}(f) = g(\|f\|_k)$ and $g : [0, \infty] \to \mathbb{R}$ is a monotonically strictly non-increasing function. Here $\|\cdot\|_k$ is the norm in $\mathcal{H}_k$. This result follows from the representer theorem [101], first introduced by Kimeldorf and Wahba [59, 60], and later extended to more general loss functions L, and regularization terms g by Schölkopf et al. [101].

Learning an estimator in an RKHS is, therefore, reduced to optimizing over a finite set of coefficients $\{\alpha_i\}_{i=1}^n$. Consequently, the computational complexity does not depend on the dimension of the feature space but rather on the number of training samples. This is advantageous when the dimension of the feature space is significantly larger than n, which is the case for many RKHS settings [101, 126]. However, in many situations, this approach results in poor scaling with the number of samples n.

Computational considerations Although kernel methods have provable theoretical advantages, they suffer from scalability issues due to high memory requirements and computational costs. Kernel methods rely on constructing an $n \times n$ kernel matrix incurring an $\mathcal{O}(n^2 D)$ cost for evaluating the kernel, where D is the dimension of the ambient space where the point cloud is embedded. Furthermore, solving for the coefficients typically reduces to solving a linear system that involves inversion of the kernel matrix, with a cost of $\mathcal{O}(n^3)$. In addition to this comes the $\mathcal{O}(n^2)$ cost of storing the kernel matrix and additional $\mathcal{O}(nD)$ operations required to evaluate the function at a sample point x.

Kernel ridge regression In the following, we consider a well-known kernel method for supervised learning, namely kernel ridge regression (KRR) [99, 104]. This learning scheme considers the minimization problem in Eq. (1.2) with a squared loss combined with the penalization term $J_\lambda(f) = \lambda \|f\|_k^2$,

$$\widehat{f}_{n,\lambda} = \operatorname*{argmin}_{f \in \mathcal{H}_k} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \|f\|_k^2 . \tag{1.9}$$

This has the benefit of giving rise to a convex optimization problem in $\mathcal{H}_k$. It can be shown that the minimizer of (1.9) is of the form

$$\widehat{f}_{n,\lambda}(x) = \sum_{i=1}^{n} \alpha_i k(x, x_i)$$

where the coefficients are given by the linear system

$$(K + \lambda n I)\alpha = y, \tag{1.10}$$

where $\alpha = (\alpha_1, \ldots, \alpha_n)$ and $y = (y_1, \ldots, y_n)^\top$.

The statistical accuracy of the KRR estimator $\widehat{f}_{n,\lambda}$ is optimal in a min-max sense as measured by the excess risk in $\mathcal{H}_k$, with $R_{\mathcal{H}_k}(\widehat{f}_{n,\lambda}) = \mathcal{O}(n^{-1/2})$ when $\lambda = n^{-1/2}$ [95]. However, solving the linear system in Eq. (1.10) requires constructing the kernel matrix K and storing it in memory, with significant costs as discussed above. Furthermore, direct inversion of $K + \lambda n I$ has a cost of $\mathcal{O}(n^3 + n^2 c_{k,D})$ operations where $c_{k,D}$ is the cost of evaluating the kernel k in the input space $\mathcal{X} \subseteq \mathbb{R}^D$.

In the first paper of this thesis, we develop a novel algorithm that improves the scalability of KRR. Several studies in the literature have been dedicated to this purpose, and we cover some of these methods in more detail in Section 5.

1.5 Boosting

Boosting is a framework for building a composite learner from a set of base learners, with significant generalization improvements over the base learners from which it is derived. First introduced by Freund and Schapire in [43, 44, 100], the boosting framework has shown great success in producing efficient learning algorithms. Following the discussion in Friedman [45, 46] we offer an overview of the boosting framework with a particular focus on gradient boosting with L_2 loss.

Consider the learning setting outlined at the beginning of Section 1. The objective is to find a function f that minimizes the expected risk $\mathbb{E}[L(f(X), Y)]$ for some loss function L, given a dataset $\mathcal{D}_n$. Boosting is similar to ensemble learning algorithms that seek to minimize $\mathbb{E}[L(f(X), Y)]$ using an estimator of the form

$$\widehat{f}_{n,\eta,\gamma}(x) = \sum_{l=1}^{T} \eta_l h(x, \gamma_l),$$

where $h(x, \gamma_l)$ is a set of base learners parameterized by γ_l. The optimal estimator is found by fitting the parameters $\{(\eta_l, \gamma_l)\}_{l=1}^{L}$ to the training data.

What sets boosting apart from other ensemble learning algorithms is the way the parameters are found. With a finite dataset $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^{n}$, boosting proposes the following iterative optimization scheme

$$(\eta^{(l)}, \gamma^{(l)}) = \underset{\eta,\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(\widehat{f}^{(l-1)}(x_i) + \eta h(x_i, \gamma), y_i) \tag{1.11}$$

where $h(x, \gamma) \in \mathcal{H}$ are base learners chosen from some hypothesis space $\mathcal{H}$. After each step, the master model $\widehat{f}^{(l)}(x)$ is updated according to $\widehat{f}^{(l)}(x) = \widehat{f}^{(l-1)}(x) + \eta^{(l)} h(x, \gamma^{(l)})$.

The advantage of this iterative approach is that each new base learner added to $\widehat{f}^{(l)}$, is exposed to the generalization error of the previous model. Consequently, new base learners can improve the performance of the estimator where improvement is most needed [40]. Furthermore, boosting is shown to work well for weak learners (simple functions that are easy to fit). The advantage is that less computational resources are required.

Gradient boosting An efficient way to approximate the optimization in Eq. (1.11), is gradient boosting proposed in Friedman [45]. The underlying idea is that the increment $\eta h(x, \gamma)$ at iteration l is a step in the hypothesis space $\mathcal{H}$. Clearly, the increment that minimizes the loss L in Eq. (1.11) the most, is in the direction best aligned with the negative gradient of L, evaluated at the current position in $\mathcal{H}$

$$g_l(x) = \left. \frac{\partial L(f(x), y)}{\partial f(x)} \right|_{f(x)=\widehat{f}^{(l-1)}(x)}$$

With finite data $\mathcal{D}_n$, the gradient direction can be estimated as $g_l = \Big(g_l(x_1), \ldots, g_l(x_n)\Big)^\top$, where $x_i \in \mathcal{D}_n$. Fitting the base learner $h(x, \gamma)$ to g_l by solving an optimization over γ gives the step direction in $\mathcal{H}$. To find the step length η_l, a new procedure can be run over values of η.

Remark 3 *We note that for the L_2 loss, an explicit expression can be found and shown to be $g_l(x_i) = y_i - \widehat{f}^{(l-1)}(x_i)$. This means that the best base learner at step l is the learner that gives the best fit to the residual after the previous step.*

In the first paper in this thesis, we develop an algorithm that combines KRR, discussed in Section 1.4, with gradient boosting. The algorithm utilizes the fact that the boosting framework works well for weak learners, which allows the kernel

at each step to be selected without too much effort dedicated to finding the optimal bandwidth. This allows the algorithm to avoid the expensive hyper-parameter tuning discussed in Section 1.2. The hyper-parameter selection strategy together with the iterative nature of boosting, allows the algorithms to work efficiently with streaming data.

1.6 Alternative methods

Learning in the non-parametric setting is not restricted to kernel methods. Other notable examples are smoothing and multi-variate splines [49], regression trees [69] and neural networks [80]. A comparison is given in Table 1.1.

Neural networks are particularly interesting due to their widespread use and great success in practical applications. In many ways, neural networks can be considered a counterpart of kernel methods; whereas learning from finite data with kernel methods is well understood in learning theory, neural networks have less theoretical support. On the other hand, kernel methods are, in their standard form, prevented from large-scale applications due to their large memory requirements and computational expenses. At the same time, deep neural networks [113] are the current go-to method for big data applications.

Model	Regression trees	Kernel methods	Neural networks
Theoretical foundation	Medium	High	Low
Interpretability	Medium	Medium	Low
Scalability	High	Low	High
Predictive power	Medium-High	High	High

Table 1.1: Comparison of different regression models. Comparison partly based on Table 10.1 in Hastie [51]. We note that although regression trees in their standard form are considered to have poor predictive power [51], boosted regression trees have proven to be very successful, and can also be combined with parallelization [119].

2 The curse of dimensionality

A fundamental challenge when learning a regression function f from a set of known training samples $\mathcal{D}_n$, is that the number of samples n needed to achieve a certain accuracy grows exponentially with the dimension of $\mathcal{X}$. This problem is often referred to as the *curse of dimensionality* [17] and occurs in several big data applications such as medicine [18], neuroscience [5] and time series [122].

We can understand the cause and implications of the curse of dimensionality from two perspectives. The first perspective is geometrical and relates to how the volume of space increases with the dimension. The other perspective comes from statistical learning theory and relates to how optimal estimation rates degrade with increasing dimensionality.

Geometric perspective Consider n samples distributed uniformly in a D-dimensional unit ball. It can be shown that the median distance from the origin of this ball to the closest data point has the following dependency on n and D [51]

$$d(n, D) = (1 - (1/2)^{1/n})^{1/D}.$$

In particular, $\lim_{D\to\infty} d(n, D) = 1$. Consequently, the distance between samples increases exponentially with the dimension (i.e. the density of samples decreases).

The implication when learning a function from a training set $\mathcal{D}_n$ is that high dimensions force extrapolation over large distances; unless we compensate with exponentially more samples. We should therefore expect function estimation to suffer in this regime.

Estimation rates perspective The geometrical perspective is useful for gaining intuition on why the curse of dimensionality occurs. However, to fully appreciate the consequences, it is useful to consider the min-max estimation rates discussed in Section 1.3, as these provide lower bounds on how well a regression function can be estimated from a training set $\mathcal{D}_n$ given a specific distribution class; see Györfi et al. [49] and Novak and Triebel [82] for more details.

For our purposes, it suffices to restate a well-known result that provides a lower bound for most regression problems in $\mathbb{R}^D$. Following Györfi et al. [49] we define the distribution $\mathcal{C}^{q,C}$ in Definition 4.

Definition 4 *Let $\mathcal{C}^{q,C}$ be the class of distributions of the random variables (X, Y), where $X \sim Uni([0,1]^D)$, $Y = f_\rho(X) + \eta$, $f_\rho \in \mathcal{F}^{q,C}$ and the noise $\eta \sim \mathcal{N}(0,1)$ is independent of X. Here $\mathcal{F}^{(q,C)}$ denote the class of all (q, C)-smooth functions $f : \mathbb{R}^D \to \mathbb{R}$, such that for $\alpha \in \mathbb{N}_0^D$ and $q = k + \beta$ we have $|\partial_\alpha f(x) - \partial_\alpha f(z)| \leq C|x - z|^\beta$, for $x, z \in \mathbb{R}^D$, $C \geq 0$, $k \in \mathbb{N}_0$, $|\alpha| \leq k$ and $0 < \beta < 1$; see Györfi et al. [49] for more details.*

It can be shown that the min-max rate for the distribution class $\mathcal{C}^{q,C}$ is

$$\inf_{f_n} \sup_{(X,Y)\in\mathcal{D}^\alpha} \mathbb{E}[(\widehat{f}(X) - f(X))^2] = \mathcal{O}(n^{-\frac{2\alpha}{2\alpha+D}}) \tag{1.12}$$

In other words, the number of samples necessary to achieve a mean square error accuracy of ε grows exponentially with D as $\varepsilon \to \infty$.

Avoiding the curse of dimensionality As we have seen, learning in high dimensions is computationally infeasible due to the large number of samples required. If data is sampled from a distribution supported in a high dimensional space $\mathbb{R}^D$ there is not much we can do. However, data is often supported on lower-dimensional subsets $\mathcal{X}$. Consequently, if this structure can be captured, significant improvements can be made. For example, when data lies on a d-dimensional linear subspace or a d dimensional smooth manifold, the optimal convergence rate from Eq. (1.12) reduces to $n^{\frac{-2\alpha}{2\alpha+d}}$. For $d \ll D$ this can make a substantial difference.

The benefit of reducing the dimensionality of the representation has motivated a vast literature on non-linear dimensionality reduction techniques (NLDR). We discuss some of these schemes in more detail in Section 4.1 with NLDR methods relevant to the work in this thesis.

3 Point clouds and their intrinsic structure

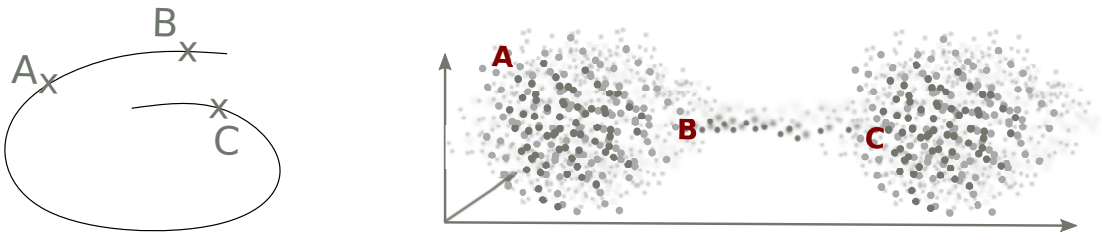
A point cloud is a collection of points sampled from a probability measure $\rho_{\mathcal{X}}$ supported on a lower dimensional set $\mathcal{X} \subset \mathbb{R}^D$, whose intrinsic structure is unknown. In practical applications, data is often modeled as point clouds instead of using more stringent assumptions such as that of a manifold; see Sindhwani et al. [108] and Little et al. [67] with references therein for examples. Furthermore, many well-known NLDR algorithms such as Laplacian eigenmaps [14], diffusion maps [31], and geometric multi-resolution analysis (GMRA) [4] have theoretical guarantees derived under assumptions of a smooth manifold, but are in practice often applied to point clouds with less structural assumptions.

It is common to characterize the intrinsic dimension of point clouds in terms of the covering dimension [84, 123] and the doubling (Assouad) dimension [1, 123].

Definition 5 (The doubling dimension) *(Adapted from Abraham et al. [1]) Let (M, d) be a metric space. The doubling dimension of M is defined as $\mathbf{ddim}(M) = \log_2(\kappa)$, where κ is the minimal number of balls of radius $r/2$, required to cover a ball $B_r(x)$ for all $x \in M$ and for all $r > 0$.*

Point clouds can have highly involved and non-linear intrinsic structures, which can impose challenges for algorithms designed to learn patterns in the data. The non-linear structure implies that the Euclidean distance induced by the ambient space $\mathbb{R}^D$ is a poor proxy for measuring distances between samples far apart. Figure 1.2a illustrates this situation. Consider the distance between the points labeled A, B, and C. Using the distance of the ambient space, the distance between A, B is larger than the distance between B, C. However, a more natural distance would be along the swiss-roll, in which case A, B is closer.

Furthermore, the density of the point cloud, encoded in $\rho_{\mathcal{X}}$ might vary across $\mathcal{X}$, and for applications such as clustering, capturing these variations in density is another aspect of importance. For example, consider the distance between A, B, and B, C on the dumbbell distribution in Figure 1.2b. When A, B belongs to the same cluster, it might be more natural that their distance should be smaller than that between B, C.



(a) Distance on swiss-roll

(b) Distances on dumbbell distribution

Figure 1.2: Illustration of how the intrinsic structure of point clouds affects what constitutes a natural distance.

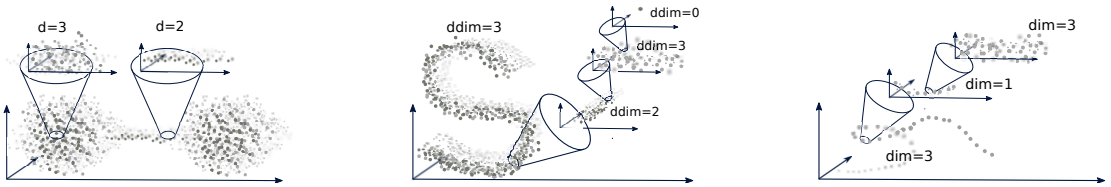
Moreover, an important aspect to consider is that the intrinsic dimension of point clouds can vary [33, 67]. Perhaps the most intuitive way that the dimension

can vary is between different regions of $\mathcal{X}$. Figure 1.3a illustrates this situation for a dumbbell-shaped point cloud where the spheres are 3-dimensional, and the connecting bridge is a 2-dimensional plane.

A somewhat less intuitive case is when the intrinsic dimension, as measured by the doubling dimension, varies with the resolution at which we consider the problem. Figure 1.3b illustrates this. With a large enough radius r when measuring the doubling dimension, the point cloud will have the dimension of the ambient space. However, with an appropriate resolution (radius), we have a more natural intrinsic dimension, namely a set that is locally approximately 2-dimensional.

The impact of resolution on the perceived dimension has implications for algorithms relying on the ambient distance metric in local neighborhoods. Examples are the kernel radius used to construct the nearest neighborhood graph in spectral graph embeddings [14, 30, 93, 115] and the radius of radial kernels used in kernel methods [104]. In other words, the radius used in these methods has implications for the structure that they see.

Another aspect associated with the resolution is the noise; when the radius is of the same order of magnitude as the noise level, the intrinsic dimension becomes that of the ambient space, as illustrated in Figure 1.3c. In general, there can be regions or scales where the dimension is very close to the ambient dimension. In this case, learning becomes practically infeasible, as discussed in Section 2. In the first paper in this thesis, we propose a strategy that can identify such regions and effectively give up when the dimension is too large, focusing instead on learning in regions where the dimension is lower.



(a) Intrinsic dimension changes with location.

(b) Intrinsic dimension changes with the resolution.

(c) Noise has high intrinsic dimension.

Figure 1.3: Three examples of variation in the intrinsic dimension. The coloring of the point clouds illustrates depth. (a) shows how the intrinsic dimension can change with the location. (b) shows how the intrinsic dimension can change with the resolution scale. (c) shows how the intrinsic dimension of noise is high-dimensional.

4 Unsupervised learning of intrinsic structure

As motivated in Section 3, learning the intrinsic structure of point clouds is of interest for reducing the impact of the curse of dimensionality and defining natural notions of similarity between samples. In the following, we consider some notable approaches relevant to the work in this thesis. In particular, this section will focus on graph-based methods and kernel PCA.

4.1 Graph methods

In graph theory [127], a graph (X_n, E) is a mathematical structure that models relationships between objects called nodes (or vertices) $X_n = \{x_1, \dots, x_n\}$ by assigning edges $E = \{(x_i, x_j) : x_i, x_j \in X_n,\, i \neq j\}$ between pairs of nodes that are connected. The most rudimentary graph has only these two properties. However, other properties are often added to model more sophisticated relationships. For example, edges are often associated with weights W_{ij} to distinguish between different degrees of similarity. The edge weights are typically represented as a matrix $W \in \mathbb{R}^{n \times n}$, called the *weight matrix*, resulting in a weighted graph (X_n, W); see Figure 1.4. Other properties to take into consideration can be the direction of edges, connectedness etc. In this thesis, we will mainly be concerned with undirected weighted graphs with symmetric edges. For discussion on other graph structures, we refer to the literature on graph theory [127].

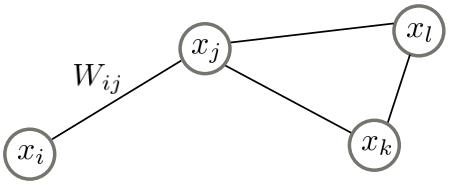


Figure 1.4: Illustration of a simple weighted graph with nodes $X_n = \{x_i, x_j, x_k, x_l\}$.

The structure of graphs makes them well-suited for modeling relationships between entities in complex systems. Examples of use range from social networks [75], biological systems [86] to computer science in general [91]. By modeling these systems as graphs, they can be analyzed efficiently and benefit from the vast literature in graph theory. An important direction in this regard is spectral graph theory [111], which studies graphs and functions on graphs through eigenvectors and eigenvalues of graph matrices. The matrices most typically studied for these purposes are the random walk matrix $A = D^{-1}W$ (also known as the diffusion matrix) and variations of the graph Laplacian $L = W - D$. Here $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix called the degree matrix where the diagonal entry $D_{ii} = \sum_j W_{ij}$ is the degree of node x_i.

In many situations, data is provided as a point cloud $\mathcal{X}$ embedded in some metric space without an explicit graph structure. Before these data sets can be analyzed using techniques from graph theory and spectral graph theory, it is necessary to first represent the point cloud as a graph.

Representing point clouds as graphs Consider a point cloud $\mathcal{X} \subset \mathbb{R}^D$ embedded in some ambient space $\mathbb{R}^D$. We can represent $\mathcal{X}$ as a graph by constructing a local neighborhood graph with edge weights $W_{ij} = k(x_i, x_j)$, for some function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Typically, k is a positive semi-definite function (a kernel function) concentrating in a local neighborhood around each point. Popular choices are the Gaussian kernel $k(x_i, x_j) = \exp(-d(x_i, x_j)^2/\sigma^2)$ and the radial kernel $k_r(x_i, x_j) = \mathbb{1}(d(x_i, x_j) \leq r)$ where $r > 0$ is some fixed radius and $d(\cdot, \cdot)$ is the metric of the ambient space.

Definition 6 (Local neighbourhood graph) *Let (M, d) be a compact metric space and let $\mathcal{X} \subseteq M$. Let $X_n = \{x_1, \ldots, x_n\} \sim \rho_{\mathcal{X}}$ be a set of points sampled from a distribution $\rho_{\mathcal{X}}$ supported on or concentrated around $\mathcal{X}$. Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a kernel function. The local neighborhood graph is the weighted graph defined by (X_n, W) where X_n are the graph nodes and $W_{ij} = k(x_i, x_j)$ are the edge weights.*

The intuition for this construction is that the neighborhood graph is a discrete approximation of the point cloud under the assumption that the distance induced by the ambient space is meaningful in small neighborhoods around each point. This is further supported by results from manifold learning [55], which is concerned with reconstructing the underlying low-dimensional structure of point clouds that lie on or close to a lower-dimensional manifold. For example, consider a point cloud $\mathcal{X} \subseteq \mathbb{R}^D$ supported on a lower-dimensional manifold. It can be shown, under certain assumptions, that the eigenvectors of the associated graph Laplacian approximate the Laplace-Beltrami operator on the manifold [13, 52]. Similarly, it can be shown that the graph geodesic, the shortest path between two nodes in a graph, converges to the geodesic on the manifold [115].

In the following, we discuss some of the ways graph methods can capture the intrinsic structure of point clouds.

Random walks and Diffusion maps Perhaps the simplest method used to capture the structure of graphs is that of the graph random walk [70]. The random walk on a graph is a process that, at each time step t transitions between graph nodes with a certain probability. In its simplest form, if at time t we are at node i, then the random walk transitions to node x_j with probability

$$p_{ij} = W_{ij}/D_{ii}$$

where W_{ij} is the edge weight between nodes x_i, x_j and D_{ii} is the degree of node x_i. The sequence of steps induced by the random walk is a Markov chain, and many properties of graph random walks can be derived from the study of Markov chains.

The random walk matrix $A = D^{-1}W$ encodes the random process, such that the i, j-th entry of A^t is the probability that we reach node j after t steps, starting at i. Let $p_t \in \mathbb{R}^n$ be a distribution on the nodes at step t. The probability of being at any node $x \in X_n = \{x_i, \ldots, x_n\}$ in the graph can then be encoded by the equation $p_t = A^t p_0$, where p_0 is the initial distribution. The stationary solution of this process is $p_{t'+1} = p_{t'} = p$, meaning that $p = Ap$. The stationary solution p is, therefore, an eigenvector of A with eigenvalue 1. It is easy to verify that p is the largest eigenvalue solution [70].

Stationary solutions of the random walk matrix have been utilized in many applications where the eigenvector entries assign a score to each node. Page rank [21, 85] uses a stationary solution of a modified Markov process, following a specific initial distribution p_0, to rank web pages. Meanwhile, label propagation [130] uses a trap or "ground" (i.e., a node with zero escape probability) to generate a stationary distribution with a decay towards nodes with a different label. For example, with

two classes A, and B, the nodes with the label A act as a ground for random walks from the nodes labeled B and vice versa.

In the fourth paper in this thesis, we utilize the stationary solution of a particular transition matrix on a nearest neighbor graph to uncover the structure in a point cloud. The transition matrix can be interpreted as a modified random walk matrix, subject to the effect of a universal ground, that imposes a termination probability to each step in the walk. This imposes a gradual decay in the probability of walking away from the source. The decay depends on the intrinsic graph structure and the "strength" of the ground.

The diffusion maps (DM) algorithm [30] is closely related to the random walk matrix. Diffusion maps provide an embedding of a point cloud into a k-dimensional Euclidean space, that allows distance between points to be calculated using the Euclidean distance in the embedding space. Diffusion maps construct a nearest neighbor graph on the point cloud in question and define a random walk matrix on this graph. It then creates an embedding from the first k eigenvectors of the t-th power of the random walk matrix. The spectral decay determines the number of eigenvectors used in the embedding. In the next section, we discuss diffusion maps in relation to spectral embeddings.

Spectral embeddings Using eigenvectors of graph matrices, spectral embeddings aim to find a lower dimensional representation of point clouds that preserve the intrinsic structure. These methods are often referred to as manifold learning methods as they derive their justification from this perspective. However, they are often applied successfully to point clouds that do not satisfy strict manifold assumptions [55].

The main application of these techniques is non-linear dimensionality reduction (NLDR) [63, 120] where they are used to overcome the curse of dimensionality, as discussed in Section 2. However, spectral embeddings have also been used successfully for several other purposes such as spectral clustering [112, 124] and semi-supervised learning [15].

Notable methods in this category are ISOMAP [115], locally linear embedding (LLE) [93], Laplacian eigenmaps (LE) [14, 16], and diffusion maps [31]. These methods all rely on a similar pipeline which involves constructing a local neighborhood graph using a PSD kernel and then defining an $n \times n$ matrix on this graph. The last step is the spectral embedding step, where the lower dimensional embedding is found by calculating the eigenvectors of the $n \times n$ matrix [55]. As an example, following Belkin and Niyogi [14] the d-dimensional LE embedding of the graph nodes X_n can be defined as

$$x_i \mapsto (v_1(x_i), \ldots, v_d(x_i))^\top \tag{1.13}$$

where $v_l : X_n \to \mathbb{R}$ is the eigenfunction corresponding to the i-th smallest eigenvalue of L. Note that the eigenfunction of $\lambda_0 = 0$ is not included as it is constant for all nodes.

The main difference between the methods lies in the matrix used for the spectral embedding. For example, ISOMAP constructs an $n \times n$ distance matrix on the graph by considering the shortest path between the nodes and then finds a

lower dimensional embedding through multi-dimensional scaling, which involves calculating the eigenvalues of this distance matrix. Meanwhile, LE relies on a spectral decomposition of the graph Laplacian, which, similarly to the ER distance, incorporates all possible paths between points [50]. Due to the instability of the shortest path distance w.r.t noise, the LE is generally more stable [14]. Furthermore, assuming that the point cloud is sampled from a manifold, the LE can be shown to converge to the corresponding Laplace-Beltrami operator [13, 52].

The major problem with spectral methods is the computational cost of the eigenfunction calculations. Furthermore, eigenfunction calculations are typically difficult to parallelize and often exhibit **global** behavior, see Definition 7, which means that all data points must be available. Another drawback of these methods is that they are incompatible with streaming, as calculating the embeddings for new data requires the entire procedure to be re-run [55].

Definition 7 (Global and local functions) *Let $f : X_n \to \mathbb{R}$ be a function defined on the nodes in the graph (X_n, W). We say f is **global** if $|f(x_i)| > \eta$, $|f(x_j)| > \eta$ for $d(x_i, x_j) > r$ for some large η. We say f is **local** if for any nodes where $d(x_i, x_j) > r$ we have that $|f(x_i)| < \zeta$ or $|f(x_j)| < \zeta$ for some sufficiently large r and sufficiently small ζ.*

In the fourth paper of this thesis, we construct an embedding using localized functions, which are cheaper to compute than global functions and can be computed independently, which allows parallelization and distribution. This scheme is also compatible with streaming data, as the localized functions can easily be extended to new samples.

Quantifying similarity between samples A fundamental problem when learning from point clouds is defining a natural notion of distance between samples. As we have seen, the metric of the ambient space is, in most cases, only suitable locally. However, this allows using a graph as a discrete approximation of the point cloud in question and then defining more suitable distances on this graph. In the following, we discuss two important distance metrics often encountered in the literature.

Perhaps the most natural distance is the graph geodesic [22], which corresponds to finding the shortest path between two nodes in a graph. The geodesic can be found using efficient algorithms such as Dijkstra's algorithm and can be shown to converge to the geodesic on the manifold as the number of samples grows to infinity [115]. However, the shortest path distance is unstable, especially when the data is not on a manifold, and small amounts of noise can dramatically affect the result [12].

An alternative to the graph geodesic is the effective resistance distance (ER) [61]. The ER between two graph nodes $x_i, x_j \in X_n$ is normally defined as

$$R_n(x_i, x_j) = (e_i - e_j)^\top L^\dagger (e_i - e_j), \tag{1.14}$$

where $L^\dagger$ is the pseudoinverse of the graph Laplacian and $e_i \in \mathbb{R}^n$ is the basis vector for node x_i, with 1 at the i-th entry and zero otherwise.

The ER distance differs from the graph geodesic by considering all possible paths between two points instead of only the shortest path. This makes ER more stable to noise and enables capturing cluster structures; namely, tightly connected regions of the graph have a smaller ER distance than loosely connected regions. The problem with ER is that it has been shown to converge to a trivial limit in the large graph limit [71, 125], which means it does not scale very well to big data applications. In the third paper in this thesis, we look into ways to overcome this issue in order to extend ER to large graphs.

4.2 Graphs as resistor networks

A resistor network (or resistor graph) is a conceptual framework that allows working with graphs using analogies to electrical circuits. A graph (X_n, W) can be thought of as an electrical network where the nodes are connected by resistors $R_{ij} = 1/W_{ij}$. It follows that a function on the graph nodes $v : X_n \to \mathbb{R}$ can be interpreted as a voltage $v(x_i)$, which induces a current through Ohm's law such that

$$v(x_i) - v(x_j) = R_{i,j}J_{i,j}, \quad \text{or alternatively} \quad J_{i,j} = W_{ij}(v(x_i) - v(x_j)). \tag{1.15}$$

Combining this with Kirchoff's current law, which states that the sum of currents entering a node i must be zero, one can define the *energy* of the voltage

$$E(v) \coloneqq \sum_{x_i,x_j \in X_n} W_{i,j}(v(x_i) - v(x_j))^2 = v^T L v. \tag{1.16}$$

The voltage function that minimizes the energy can be used to uncover information about the graph. However, in an unconstrained system, the minimizer is trivial as zero energy can be obtained for any voltage function for which all entries are equal, namely $v(x_i) = v$ for all $x_i \in X_n$. Consequently, it is necessary to introduce constraints on the system. This is normally done by imposing conditions on the voltage or the current flow in the system. Several graph embeddings can be shown to correspond to minimizers of the energy under different constraints. As examples, we consider Laplacian eigenmaps and effective resistance.

Laplacian eigenmaps Following Belkin and Niyogi [14], the coordinate functions $\{v_l\}_{l=1}^d$ of the d-dimensional LE embedding in Eq. (1.13) can be shown to be the solutions of the energy minimization problem

$$\begin{aligned} \min_{v_l : X_n \to \mathbb{R}} \quad & \sum_{x_i,x_j=1}^{n} W_{ij}(v_l(x_i) - v_l(x_j))^2 \\ \text{subject to} \quad & v_l \perp v_{l'} \quad \text{and} \quad v_l \perp 1. \end{aligned} \tag{1.17}$$

for $l = 1, \ldots, d$. The constraint $v_l \perp \mathbf{1}$ is introduced to avoid the trivial solution while $v_l \perp v_{l'}$ ensures that the voltage solutions are orthogonal. Here $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones. Meanwhile, it is worth noting that there are no constraints imposing any locality of the voltage functions. This explains why the eigenfunctions of LE can often be global, as discussed in Section 4.1. Furthermore,

the orthogonality condition prevents calculating the eigenfunctions independently. Consequently, this limits the use of distribution and parallelization to solve the minimization problems.

In the fourth paper in this thesis, we address this by introducing constraints that force the voltage function to be local, in the sense of Definition 7. Furthermore, we do not require individual voltage functions to be orthogonal, which allows them to be solved for independently.

Effective resistance Similarly to LE, effective resistance can also be formulated as an energy minimization problem. Following Jørgensen et al. [58] the ER between two graph nodes $x_i, x_j \in X_n$ can be defined as in Proposition 8.

Proposition 8 *The effective resistance between nodes x_i, x_j corresponds to $R_n(x_i, x_j) = 1/J_{tot}$, where*

$$J_{tot} \coloneqq \sum_{j \in \{1,\dots n\}} W_{ij}(v^*(x_i) - v^*(x_j))$$

and v^* is the function that minimizes the Dirichlet energy

$$\begin{aligned} \min_{v: X_n \to \mathbb{R}} \quad & \sum_{x_i, x_j \in X_n} W_{i,j}(v(x_i) - v(x_j))^2, \\ \textit{subject to} \quad & v(x_i) = 1, \quad v(x_j) = 0 \end{aligned}$$

Proof See Theorem 4.2, in Jørgensen and Pearse [58]. ■

In Proposition 8, the solution is constrained using Dirichlet conditions on the voltage. However, several equivalent formulations exist, as shown in Jørgensen and Pearse [58]. Solving a minimization problem with constraints on the current, it is easy to show that the solution corresponds to the ER defined in Eq. (1.14).

4.3 Kernel methods revisited

Kernel methods and learning in RKHSs are not restricted to regression and supervised learning. Since $\mathcal{H}_k$ is a linear space equipped with an inner product, the distance between features $\phi_i \in \mathcal{H}_k$ is straightforward to compute. At the same time, from Eq. (1.6) it follows that the kernel function can be thought of as a non-linear similarity measure on the original set $\mathcal{X}$. Consequently, the kernel embedding allows samples $x_i, x_j \in \mathcal{X}$ to be compared implicitly through the inner product in $\mathcal{H}_k$, without knowing a natural notion of distance in the original space $\mathcal{X}$. In the following, we discuss a method that utilizes these observations to extend traditional PCA to non-linear structures.

Kernel principal component analysis A notable kernel method for unsupervised learning is kernel principal component analysis (kernel PCA) [102, 103]. Kernel PCA generalizes standard PCA to point clouds with non-linear intrinsic structures. The intuition is that the linear structure of $\mathcal{H}_k$ allows standard PCA to be applied.

After centering the data in the feature space, it can be shown that the p-th principal component is given as

$$v_p(x) = \frac{1}{\sqrt{\lambda_p}} \sum_{i=1}^{n} \alpha_{p,i} k(x, x_i),$$

where (λ_p, α_p) is the p-th eigenvalue-eigenvector pair of the kernel matrix $K_{ji} = k(x_i, x_j)$, and

$$K\alpha = m\lambda\alpha.$$

We see that kernel PCA, similarly to KRR, is reduced to solving a linear system, and the solution is expressed as a linear expansion in the kernel evaluated at the training samples. We note that, as stated by Schölkopf et al. [101], kernel PCA can also be thought of as a minimization problem on the form (1.2).

It is clear that kernel PCA suffers from the same scalability issues as KRR. Solving the linear system requires calculating the eigenvalues of the kernel matrix, which in general, requires $\mathcal{O}(n^3)$ operations. Furthermore, kernel PCA requires $\mathcal{O}(n^2)$ memory to store the kernel matrix and needs $\mathcal{O}(nD)$ operations to project new samples x onto the principal components.

We mention that kernel PCA gives an elegant connection between kernel methods and the spectral embedding methods discussed in Section 4.1. In particular, it is shown by Ham et al. [50] that ISOMAP, LLE, and the Laplacian eigenmaps can all be interpreted as kernel PCA for a particular choice of kernel. For example, Laplacian eigenmaps can be considered as performing kernel PCA with the pseudoinverse of the graph Laplacian, which is closely connected to commute times and the ER distance on graphs.

4.4 Alternative methods

The task of learning intrinsic structures in point clouds has been addressed in several other works in the literature, beyond kernel methods and graph-based methods. A notable approach is the geometric multi-resolution analysis (GMRA) developed in a series of works [4, 64, 65, 72]. Another notable example is local tangent space alignment [131].

5 Scaling kernel-based learning for big data

For most real-world applications, it is necessary to use algorithms that can handle large amounts of data in a resource-efficient manner. Furthermore, the scale of modern data sets has motivated the development of powerful computational models such as streaming, parallelization, and distributed systems. To fully utilize the potential of these computational models, it is essential to develop algorithms that can operate and work with data in the way the computational models demand.

As we have seen in Section 1.4 and 4.1, graph-based methods such as spectral embeddings and kernel methods such as KRR and kernel PCA are, in their basic form, prevented from large-scale applications due to their considerable memory requirements and computational costs. In particular, we have seen that these

methods rely on the construction of large $n \times n$ kernel matrices and on expensive inversions and spectral decompositions of these matrices. Furthermore, these methods are not optimized for powerful computational models such as streaming, parallelization, and distribution.

In the following, we start by reviewing the requirements on data handling imposed by computational models such as streaming, parallelization, and distribution. We then review several techniques rooted in randomized numerical linear algebra [76] used to reduce the size of kernel matrices for the purpose of improving time and memory usage. We also review iterative approaches for efficiently solving eigenvalue problems and matrix inversion problems.

5.1 Computational models

Modern data sets are often prevented from loading into memory in their entirety due to their sheer size or because they are provided only through a continuous stream of examples. Furthermore, computational models such as the streaming model of computation, parallelization, and distribution enforce their own requirements on how data can be managed. To fully utilize the potential of parallelization and distribution and to enable learning from data streams, it is necessary to be aware of the requirements of these computational models when designing learning algorithms.

In the following, we will give a brief overview of these computational models and their requirements.

Streaming model of computation A streaming model of computation [81] is necessary for data prevented from being made available in memory in its entirety. This could be because the data is too large to be kept in memory and, therefore, must be loaded incrementally or in batches. Alternatively, it could be because the data is recorded and made available continuously. The development of learning algorithms to handle streaming data is of great interest in machine learning, as shown by recent reviews by Gomes et al. [47] and Bahri et al. [10], due to the rapid increase of data exhibiting such requirements in big data applications.

Streaming algorithms read data as a single sample or a mini-batch at a time and incorporate it into the learning model. After processing a sample, the algorithm discards it to limit the data kept *in-memory*. Because of the large data size, possibly infinite, the computational complexity of operations performed in-memory can not scale with the data size. In general, this is handled by storing only a sketch of the data in-memory. The size of the sketch is usually significantly smaller or even independent of the size of the data itself. Updates to the learning models either happen incrementally with each new sample, or batch-incrementally with a batch of new samples [10].

In the first paper of this thesis, we develop a learning scheme for KRR that is designed for the streaming setting.

Parallelization and distributed computations Parallelization and distributed computations are closely linked computational concepts but differ in some

vital aspects [11]. The fundamental difference between the two is that parallel computing utilizes several processors typically sharing memory on the same computer. Meanwhile, distribution refers to computations performed at independent computers often provided through cloud services that do not have access to the same memory. Parallelization can be used to greatly increase the utilization of computer resources, while distribution allows the use of clusters to significantly scale the available resources beyond one computer.

In the fourth paper in this thesis, we develop an embedding scheme based on localized functions that can be calculated independently and do not require access to all data simultaneously, therefore allowing parallelization and distribution.

5.2 Matrix approximation techniques

When dealing with large positive semi-definite matrices, a common strategy for reducing the computational expense is to find a low-rank approximation that can replace the original matrix. This is part of a more general question, closely studied in numerical linear algebra [76], on how to find a good spanning subset of rows or columns for a given matrix. The motivation is that many matrices have singular values that decay fast. Therefore, in principle, it should be possible to approximate such matrices by a subset of basis vectors. The best rank-one approximation is clearly the leading singular vector, but as the main goal is to speed up computations, using the singular vectors is often not an option.

In numerical linear algebra, there are several methods designed for low-rank approximations of matrices. A discussion on some of these techniques can be found in Mahoney et al. [74] and in Bach [9] with particular emphasis on kernel methods. However, we will concentrate on two specific approximation techniques, namely Nyström sub-sampling [128] and random features [90], which have been particularly successful in the context of kernel methods. We mention that several other matrix approximation strategies exist. An example is block kernel approximation, which utilizes the clustering structure of kernel matrices to approximate the matrix. Another notable example is memory-efficient kernel approximation, which utilizes low-rank structures in the matrix in addition to the cluster structure to enhance the approximation further [107].

Nyström sub-sampling The main idea of the Nyström sub-sampling is to create an approximation of a PSD matrix $A \in \mathbb{R}^{n \times n}$ using a subset $\widetilde{\Gamma} \subset \Gamma = \{1, \dots n\}$ of the matrix column indices. The Nyström approximation was proposed simultaneously by Williams and Seeger [128] and Smola [109], the main difference being the column selection strategy. In its fundamental form, the Nyström approximation can be written as

$$\widetilde{A}_{nn} = A_{nm} A_{mm}^{\dagger} A_{mn},$$

where $A_{nm} = A(\Gamma, \widetilde{\Gamma}) \in \mathbb{R}^{n \times m}$, and similarly $A_{mm} = A(\widetilde{\Gamma}, \widetilde{\Gamma}) \in \mathbb{R}^{m \times m}$, are constructed on the selected subset of columns. We note that the error in the approximation is elegantly connected to the Schur complement as $A_{nn}/A_{mm} = A_{nn} - A_{nm} A_{mm}^{\dagger} A_{mn}$.

The use of the Nyström approximation is useful in many applications involving spectral decompositions and inversion of A. A typical application is where the inversion of $A + \lambda I$ is necessary. For example, KRR requires solving a linear system of the form $(K + \lambda I)\alpha = y$, where $K \in \mathbb{R}^{n \times n}$ is the kernel matrix and $y \in \mathbb{R}^n$ is the labeled training samples. Using the Nyström approximation and the Woodbury formula we get

$$\alpha = \frac{1}{\lambda}\Big(y - K_{nm}(\lambda K_{mm} + K_{mn}^\top)K_{nm})K_{mn}y\Big),$$

which can be solved with $\mathcal{O}(m^2 n)$ computations and $\mathcal{O}(nm)$ memory [128]. Other more advanced solution techniques have also been used, the most notable in terms of KRR is FALKON developed by Rudi et al. [95], which combines sub-sampling with an iterative solver and a sub-sampled preconditioner.

In the context of kernel methods, each column corresponds to a feature vector $\phi_i = k(\Gamma, x_i)$. The column sub-sampling can therefore be thought of as selecting a smaller hypothesis space $\mathcal{H}_m \subseteq \mathcal{H}_n \subseteq \mathcal{H}_k$, spanned by the features associated with the selected columns. This is similar to the restriction introduced by $\mathcal{H}_n$ in Eq. (1.8). Furthermore, since each column is associated with a specific sample in the original domain $\mathcal{X}$, the column sub-sampling can be thought of as selecting a subset of the training samples $\widetilde{\Gamma} = \{\widetilde{x}_i\}_{i=1}^m \subset \mathcal{D}_n$, often referred to as Nyström centers. The estimator is expressed as a linear expansion in the kernels centered at these Nyström centers

$$\widehat{f} = \sum_{\widetilde{x}_i \in \widetilde{\Gamma}} \widetilde{\alpha}_i k(\cdot, \widetilde{x}_i) \in \mathcal{H}_m.$$

Algorithms based on the Nyström approach differ in the way they select the subset of columns. In the following, we review some popular selection schemes.

Randomized sub-sampling: The approach normally associated with Nyström approximation is random sub-sampling, where sub-samples are selected uniformly at random without replacement. This is the original strategy proposed by Williams and Seeger [128]. Despite its simple nature, it has proven to provide good approximations and is especially attractive as it requires no extra computations and is easy to analyze theoretically [9, 94, 95]. We note that random sub-sampling is often referred to as Nyström sub-sampling, although the Nyström approximation is not restricted to the random sub-sampling choice.

A problem with random sub-sampling is that important columns can be missed. For example in regions with few available samples, random sub-sampling might miss out on sampling columns from these regions entirely. Meanwhile, regions with a high density of samples can have too much influence.

Leverage scores sub-sampling: Mahoney and Drineas [73] introduced the concept of leverage scores as a way to ensure that important columns are sampled, whereby important we mean columns that have a proportionally large effect on the low-rank fit. The fundamental idea is to create a probability distribution that reflects the importance of columns and then sample the columns according to this distribution. Alaoui and Mahoney [3] extended this concept to kernel ridge

regression, introducing leverage scores tailored to this setting. In this formulation, each training sample $x_i \in \mathcal{D}_n$ is associated with a leverage score

$$l(x_i) = (K(K + tnI)^{-1})_{ii} \quad \text{and probability} \quad p_i = l(x_i)/\sum_{i=1}^{n} l(x_i).$$

The columns are sampled with probability $p = (p_1, \ldots, p_n)$. The challenge with leverage scores is that they are expensive to compute, therefore approximations are typically used instead [29, 38, 94–96]. An efficient algorithm for approximating leverage scores can be found in Rudi et al. [96].

Non-probabilistic sub-sampling: The sub-sampling schemes reviewed above rely on a probabilistic approach to sub-sampling. However, sub-sampling strategies for low-rank matrix approximation are not limited to this setting. In particular, Smola [109] proposed a greedy strategy, relying on the pivoted Cholesky method, which iteratively searches for optimal columns. Meanwhile, Fine and Scheinberg [41] proposed a method based on incomplete Cholesky factorization. These non-probabilistic techniques give better approximations than their probabilistic counterparts but have in general larger computational expenses and are harder to analyze [9, 76].

Random features The idea behind random feature approximation of kernel matrices is rooted in the *empirical approximation method* found in approximation theory and randomized linear algebra [36, 37, 76]. The main idea is as follows, assume that we have a low-rank random matrix $Z \sim \rho_Z$ sampled from some distribution ρ_Z such that its expectation equals the matrix we want to approximate, namely $A = \mathbb{E}[Z]$. It follows that the empirical mean is a good estimator of A

$$\tilde{Z}_m = \frac{1}{m}\sum_{i=1}^{m} Z_i,$$

where each Z_i is sampled i.i.d from ρ_Z.

In Rahimi and Recht [90], this approximation strategy was introduced for kernel matrices along with the concept of random features. Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a kernel function, and let $K \in \mathbb{R}^{n\times n}$ with $K_{ij} = k(x_i, x_j)$ be the associated kernel matrix constructed on the data set $\mathcal{D}_n = \{x_i\}_{i=1}^n$. The random feature approach is concerned with finding a bounded function $z(x, w) : \mathcal{X} \times \mathcal{W} \to \{z \in \mathbb{C}, \|z\| \leq \infty\}$, sampled according to some distribution ρ_W defined on $\mathcal{W}$, such that

$$k(x_i, x_j) = \int z(x_i, w)z(x_j, w)^* d\rho_{\mathcal{W}} = \mathbb{E}[z(x_i, w)z(x_j, w)^*].$$

If $z(x, w)$ is known, the kernel matrix can be approximated by forming the random matrix $Z(w) = z(w)z(w)^*$, whose expectation is $K = \mathbb{E}[Z]$. Here $z(w) = \Big(z(x_1, w), \ \ldots, \ z(x_1, w)\Big)$ is called a *random feature*. The existence of a function $z(x, w)$ that satisfies this property is the case for several PSD kernels. For example, for the Gaussian kernel, we have $z(x, w) = \exp\left(i\langle x, w\rangle\right)$ where $w \sim \mathcal{N}(0, \sigma^{-2})$ [76]. The cost of this approximation is $\mathcal{O}(nmd)$ where d is the dimension of $\mathcal{X}$.

Computational remarks When using the approximation techniques described above, in the context of KRR, it is of interest to characterize the number of sub-samples m necessary to maintain the statistical accuracy of standard KRR.

Consider a hypothesis space $\mathcal{H}$ and the optimal statistical accuracy measured in terms of the excess risk in $\mathcal{H}$, namely $\mathcal{R}_{\mathcal{H}}(f)$ as defined in Eq. (1.4). The standard KRR estimator $\widehat{f}_n$ from Eq. (1.9), achieves the rate $\mathcal{R}_{\mathcal{H}}(\widehat{f}_n) = \mathcal{O}(n^{-1/2})$ when $\lambda = n^{-1/2}$, which is optimal statistical accuracy in a min-max sense according to Rudi et al. [95]. Table 1.2 (adapted from Rudi et al. [95]) compares this to alternative implementations of KRR for parameter choices that give the same optimal statistical accuracy.

KRR solver	Training time	Kernel evaluations	Memory	Test time
Standard KRR	n^3	n^2	n^2	n
Random features [90]	n^2	$n\sqrt{n}$	n	$\sqrt{n}$
Nyström [109, 128]	n^2	$n\sqrt{n}$	n	$\sqrt{n}$
Nyström iterative [24]	n^2	$n\sqrt{n}$	n	$\sqrt{n}$
FALKON [95]	$n\sqrt{n}$	$n\sqrt{n}$	n	$\sqrt{n}$

Table 1.2: Table adapted from Table 1 in Rudi et al. [95]. Computational and memory requirements to achieve optimal learning rates for KRR

We note that further computational improvements to KRR can be achieved by introducing iterative methods, such as gradient descent with early stopping, to approximate the solution of $(K + \lambda nI)\alpha = y$ from Eq. (1.10) [24]. FALKON proposed in Rudi et al. [95], takes this a step further by introducing a preconditioner also subject to Nyström-based sub-sampling, which further reduces computational and memory requirements while maintaining the same optimal statistical accuracy.

Utilizing GPU acceleration and parallelization, Meanti et al. [78] demonstrate that FALKON can be applied efficiently to large-scale datasets with billions of points. However, despite its success in application to large-scale data sets, FALKON still requires the selection of an optimal bandwidth and does not incorporate an efficient way to select this bandwidth in a streaming framework, where cross-validation techniques are inapplicable. The same can be said about Nyström sub-sampling techniques based on probabilistic schemes such as random sub-sampling and leverage scores, which require access to the available data to determine the optimal number of sub-samples m, which again might depend on the bandwidth. FALKON does not address the issue of selecting these samples in a streaming environment.

In the first paper in this thesis, we develop a novel algorithm utilizing a modified version of FALKON as the core solver. The algorithm introduces a novel sub-sampling and bandwidth selection scheme to extend the KRR approach to streaming data. Another notable effort to extend KRR to the streaming setting is the multi-kernel online learning scheme proposed by Shen et al. [106]. This algorithm utilizes a random feature-based approach for matrix approximation and

combines this with an iterative gradient descent-based update of the expansion weights.

5.3 Sparse eigensolvers

Finding the eigenvalues and eigenvectors of large matrices is an expensive operation. For $n \times n$ matrices, the cost is $\mathcal{O}(n^3)$. However, when additional information is known about the matrix, this cost can be significantly reduced. For example, utilizing sparsity in a matrix can reduce both memory and computational requirements. In this section, we discuss how sparsity can reduce expenses involved with finding the eigenvector of the leading eigenvalue, in terms of the power method.

Sparse matrices An $m \times n$ matrix is said to be sparse if it has $\mathcal{O}(\min(m, n))$ non-zero elements [116]. Meanwhile, a matrix that has very few non-zero elements is referred to as dense. In the modern era, sparse representation of matrices is available in most programming languages, and the advantage of working with these representations is as follows:

- *Memory*: Large matrices can be stored in a compressed form where only the non-zero elements with their associated indices are stored.
- *Computational*: Time is saved if only operations with non-zero elements are performed.

An important application of sparse representations is the power method and finding the largest eigenvector of sparse matrices.

Power method The power method [76], also known as power iteration, is an iterative method in the family of Krylov subspace methods, for finding the leading eigenvalue λ_{max} or eigenvector of a positive semi-definite matrix. Let $A \in \mathbb{R}^{n \times n}$ be a PSD matrix. Starting with an initial guess $v_0 \in \mathbb{R}^n$ the power method generates a sequence of eigenvector estimates

$$v_t = \frac{Av_{t-1}}{\|Av_{t-1}\|_2}, \quad \text{with associated eigenvalue estimates} \quad \eta_t = v_t^\top A v_t,$$

for $t \geq 1$. We note that this is similar to the procedure of finding the stationary solution of the random walk matrix, discussed in Section 4.1.

For each iteration t, the power method calculates the matrix-vector product Av_t, if the matrices are sparse, the computational cost of these operations can be significantly reduced. Further improvements can be made if the vector v_t is also sparse [8]. However, despite this, there is still the need to construct and index a large $\mathbb{R}^{n \times n}$ sparse matrix.

In the fourth paper in this thesis, we suggest a scheme that ensures that the vector v_t is spatially localized in the underlying graph, see Definition 7. Consequently, it is possible to work with only a sub-matrix of A. For large sample sizes n, this can have significant improvements as one only needs to construct the matrix on a subset of the data.

6 Manuscript contributions

In this section, we present four original manuscripts developed in this thesis. The first two manuscripts are concerned with regression in a supervised setting. The last two manuscripts are concerned with uncovering the intrinsic structure of point clouds in an unsupervised setting. The first paper presents a novel KRR solver StreaMRAK. The second paper demonstrates StreaMRAK as a tool for predicting ionic membrane currents from cardiac action potential traces. The third paper proposes a new definition of effective resistance to alleviate the convergence issues encountered by the standard definition. The fourth paper demonstrates a new embedding strategy for point clouds.

6.1 Paper I: StreaMRAK

Kernel ridge regression allows the learning of highly non-linear functions. The success of this method has been demonstrated in many applications and is supported by a well-established theoretical foundation [54, 92, 101, 104]. However, KRR, like other kernel-based learning algorithms, suffers from large memory requirements and high computational costs. These costs arise because learning with KRR involves solving a linear system $(K + \lambda nI)\alpha = y$ for the coefficients $\alpha \in \mathbb{R}^n$, where the kernel matrix $K \in \mathbb{R}^{n \times n}$ grows with the number of samples.

Efforts to overcome computational expenses and large memory requirements have focused on reducing the size of the kernel matrix with sub-sampling techniques such as Nyström approximations and random features. Furthermore, using scalable iterative methods to solve the linear system $(K + \lambda nI)\alpha = y$ have been shown to significantly cut down the computational costs. These efforts have led to many capable KRR solvers, as summarized in Table 1.2. A prominent example is FALKON, developed by Rudi et al. [95], which has been demonstrated to work efficiently with massive datasets [78].

Manuscript contribution We develop a novel kernel-based learning algorithm called StreaMRAK, for the purpose of extending KRR to the streaming computational model. The contributions of this algorithm can be summarized as

1. Efficient use of samples.
2. Efficient selection of hyper-parameters.
3. Compatibility with streaming.
4. A novel way to mitigate the curse of dimensionality.

The motivation for the algorithm is that in a streaming computational model, we expect large amounts of data to arrive sequentially or in batches. However, computational resources and memory are limited. In light of this, StreaMRAK has been designed to make efficient use of samples, only storing samples as long as they are needed and then discarding them.

Furthermore, the use of cross-validation to optimize the kernel bandwidth is cumbersome in a streaming setting. Therefore, StreaMRAK implements a multi-resolution approach to learning that consists of two parts, a novel sub-sampling scheme combined with an efficient bandwidth selection strategy. Together, these methods adapt the bandwidth and sub-sample density to the resolution level in a data-driven manner. The benefit is that expensive optimization over the bandwidth hyper-parameter is avoided, although at the cost of not finding the best possible bandwidth at each level. The sub-sampling part of the multi-resolution approach is formulated as a pyramid, starting at a low-resolution level l, with few sub-samples from the data, it gradually increases the resolution and number of sub-samples for growing l.

The multi-resolution scheme also includes a boosting formulation of KRR, where the estimator at level l is defined as

$$\widehat{f}_{n,\lambda}^{(l)}(x) = \widehat{f}_{n,\lambda}^{(l-1)}(x) + \widehat{s}_{n,\lambda}^{(l)}(x).$$

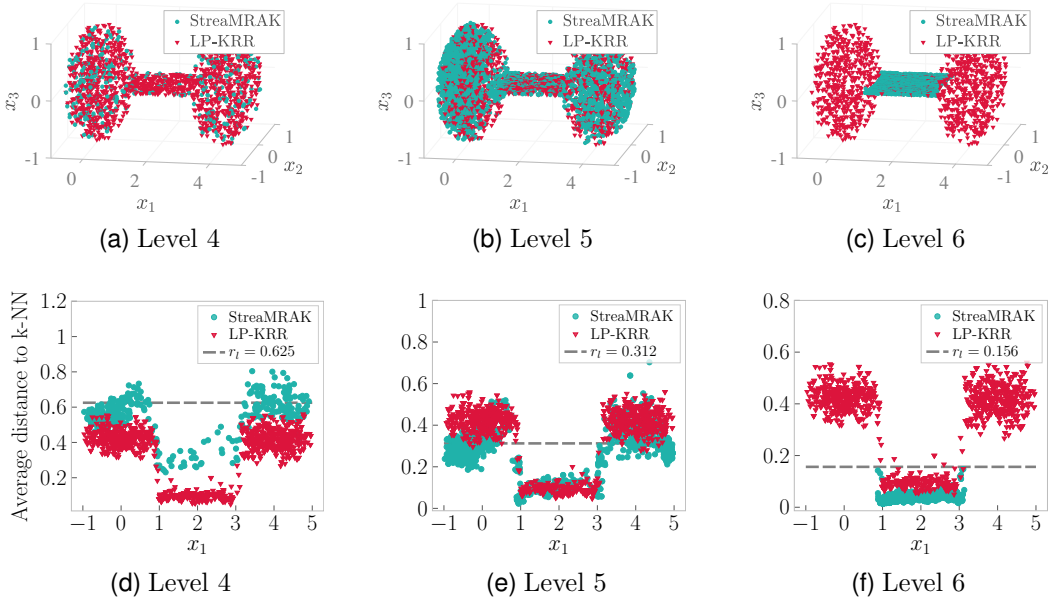
and $\widehat{s}_{n,\lambda}^{(l)}$ is the estimator obtained after regression on the residual $d^{(l)}([x_n]) = y - \widehat{f}_{n,\lambda}^{(l-1)}([x_n])$, where $d^{(0)} = y$. FALKON is employed as a base solver to solve the KRR at each level. Here we take $f([x_n])$ to mean $f([x_n]) = (f(x_1), f(x_2), \ldots, f(x_n))^\top$.

This procedure corresponds to gradient boosting with L_2 loss from Section 1.5, where the step length is $\beta^{(1)} = 1$ at each step. Specific to StreaMRAK is that the samples $[x_n]$ used to calculate the residual at each level, are sampled independently from the samples used to train the model at the previous level.

We note that the multi-resolution scheme developed in StreaMRAK is inspired by a specific multi-resolution scheme used in image analysis, known as the Laplacian pyramid (LP) [23]. Moreover, during a further literature review, after finalizing the paper, we were able to establish a connection between the LP and a particular version of gradient boosting. In fact, Shao et al. [66] proposed a boosted version of KRR similar to StreaMRAK. However, we note that the boosted KRR developed in Shao et al. [66] does not correspond to a multi-resolution scheme. This is because the bandwidth is kept fixed at each level and the sub-sampling density is not adapted to the bandwidth. Regardless, the boosting perspective provides a useful foundation for interpreting the performance of StreaMRAK, and is, therefore, the perspective we have chosen to take in this discussion.

Boosting achieves two things. First, it is known to generate a composite estimator with better generalization properties than its base learners, even when these are weak learners. This justifies the adaptive bandwidth-selection approach discussed earlier. Since weak learners are acceptable, kernels can be defined with a bandwidth that is adequate without too much effort dedicated to finding the optimal one. Secondly, since StreaMRAK uses new samples at each level to evaluate the residuals $d^{(l)}([x_n])$, the estimator at the next level sees the generalization error of the previous and can compensate for it.

Finally, learning in high dimensions is known to be infeasible due to the large number of samples required; see Section 2. This problem is especially problematic in a multi-resolution scheme since high resolution, i.e. small kernel bandwidth, requires a high density of samples. Furthermore, as discussed in Section 3, the



(a) Level 4 (b) Level 5 (c) Level 6

(d) Level 4 (e) Level 5 (f) Level 6

Figure 1.5: Demonstration of how StreaMRAK adapts the sub-sampling density to the kernel bandwidth. The upper row shows the distribution of sub-samples on the dumbbell. The lower row shows the average distance to the 7-nn samples. The sub-samples selected by StreaMRAK are marked in blue, while the samples selected by random Nyström sub-sampling are marked in red. The grey dotted line in the lower row is the bandwidth. (Here LP-KRR refers to sub-sampling using random Nyström sub-sampling)

intrinsic dimension of point clouds can change based on location and resolution. In particular, if the noise level is reached, the intrinsic dimension becomes that of the ambient space. Because of this, StreaMRAK implements a scheme to identify regions of high dimensionality. The strategy is then to dedicate fewer resources to these regions and focus instead on lower-dimensional regions where learning is feasible.

To better understand the adaptive bandwidth- and sub-sampling strategy, we refer to Figure 1.5, which illustrates the method on a dumbbell-shaped point cloud. In the dumbbell, the spheres are 5-dimensional, and the connecting plane is 2-dimensional. In the figure, the upper row shows how the sub-samples are distributed on the dumbbell for levels $l = \{4, 5, 6\}$. The blue marks refer to sub-sampling with StreaMRAK, while the red marks (labeled LP-KRR) refer to random Nyström sub-sampling. The lower row in the figure shows, for levels $l = \{4, 5, 6\}$, the average distance between each sub-sample and its 7-nearest neighbors in the set of sub-samples. These distances are compared with the kernel bandwidth r_l at the corresponding level, shown by the grey dotted line. At each level, the kernel bandwidth is reduced by $r_l = 2^{-l} r_0$ from some initial bandwidth $r_0 > 0$. It is desirable that the average distance between sub-samples is comparable in magnitude with the kernel bandwidth. It is apparent that the sub-sampling used in StreaMRAK, adapts better to the bandwidth. In addition, from Figure 1.5f, we can see how StreaMRAK gives up in high dimensions by no longer selecting samples from the spherical regions when the bandwidth becomes too small.

6.2 Paper II: Improving inversion of model parameters from action potential recordings with kernel methods

An important aspect of developing anti-arrhythmic cardiac drugs is the measurement of ionic membrane currents $p = (p_1, \ldots, p_d)$ in cardiomyocytes. These ionic currents are responsible for the electrical properties and dynamics of cardiomyocytes, which in turn are essential to the contractions generated by these cells; see Remark 9. Furthermore, most anti-arrhythmic drug agents interact with ionic channels in the cellular and sub-cellular membranes to modulate ionic currents. Consequently, the measurement of ionic membrane currents can be used to guide the development of drugs that target these channels and give valuable insights into heart disease and electrical properties of the heart.

Remark 9 *"Cardiomyocytes are the cells responsible for generating contractile force in the intact heart."[129]*

Direct measurements of ionic membrane currents require expensive equipment and specialized practitioners [26, 62]. Meanwhile, the dynamics of these currents are responsible for generating the cardiac transmembrane potential v, known as the action potential (AP). We denote this relationship as $v = f(p)$. The AP can be measured at significantly lower cost and expertise, using techniques such as live cell fluorescence microscopy and microelectrode arrays [28, 53, 77]. Furthermore, several mathematical models $\widetilde{f}$ are developed to approximate the function f [39, 42, 48, 83, 87, 89, 97, 98, 114, 117]. Because of this, AP measurements, together with AP models, are a promising gateway to efficiently quantifying ionic membrane currents.

Problem 10 *Given an experimentally measured AP trace $w_i = (w_{i1}, \ldots, w_{iT}) \in \mathcal{V}_T \subset \mathbb{R}^T$ where T is the number of recorded time steps. Characterize the corresponding ionic membrane currents $p = (p_1, \ldots, p_d) \in \mathcal{P} \subset \mathbb{R}^d$ with the help of an AP model $\widetilde{f} : \mathcal{P} \to \mathcal{V}_T$.*

Problem 10 is an inverse problem. Namely, given a set of observations we want to find the parameters that caused them. This problem is, therefore, often referred to as the problem of AP trace inversion. What makes Problem 10 challenging is that the relationship between ionic membrane currents and the cardiac action potential is highly non-linear and stochastic in nature [88]. Furthermore, the AP is determined by substantial amounts of distinct ionic currents, many of which are of interest to identify in clinical applications. The AP models, designed to capture these dynamics, inevitably consist of complex systems of equations, typically large systems of ODEs that are expensive to compute; see e.g. Qu et al. [88] for a review on AP models and their construction.

Moreover, in many AP models, several ionic currents that are of interest to identify suffer from sensitivity and identifiability issues [56]. Meaning that their effect on the AP is hard to detect or the effects from different currents cancel each other out in certain regions of the parameter domain. Consequently, when designing algorithms for AP trace inversion, these are challenges that need to be taken into account.

In the literature on AP trace inversion, Problem 10 is normally addressed by defining a loss function $L(\phi^{AP}(w), \phi^{AP}(v))$ over a set of AP features $\phi^{AP}(v) = (\phi_1^{AP}(v), \ldots, \phi_m^{AP}(v))$, $\phi_i : \mathcal{V}_T \to \mathbb{R}$ constructed on the AP traces. The strategy is to search in parameter space $\mathcal{P}$ for a parameter vector p whose corresponding AP trace $v = \widetilde{f}(p)$ minimize the loss function. To find the minimizer, the common strategy is to use gradient-free iterative optimization schemes such as Nealder-Mead and Particle swarm [27, 57, 68]. However, the challenge with these iterative optimization schemes is that they must solve the AP model at each iteration. Since existing AP models are large systems of ODEs that are expensive to solve, this makes iterative optimization schemes slow in the face of large datasets.

In Tveito et al. [118], this scalability issue is addressed by first sampling a large quantity of ionic current parameters $\{p_i\}_{i=1}^n$ from $\mathcal{P}$, either uniformly or from a grid. The system of ODEs is then solved on these parameters to generate a dataset $\mathcal{D}_n = \{(v_i, p_i)\}_{i=1}^n$ consisting of AP traces and the corresponding current parameters. We refer to this dataset as a "pre-computed" dataset. For a given measured AP trace w, one can then search for the closest AP trace within $\mathcal{D}_n$, namely $v_{opt} = \text{argmin}_{v \in \mathcal{D}_n} L(v, w)$, where L is some loss function. If n is small, v_{opt} can be found by brute force, computing the distance between all sample pairs. However, for sufficiently large n, this is not computationally viable. In Tveito et al. [118], an iterative scheme, searching in bounding boxes defined in $\mathcal{P}$, was used instead. Thereby reducing the number of samples to compare.

The advantage of using a pre-computed dataset is that it moves the computational expense to a pre-computation step, making the algorithm significantly faster in the prediction phase; where one wish to find the ionic membrane currents corresponding to AP traces w measured in the lab. However, this comes at the cost of introducing large memory requirements in storing the pre-computed data, as well as requiring advanced methods for reading and accessing the data.

Manuscript contribution In this paper, we propose solving Problem 10 by learning an estimator $\widehat{f}_n$ of the inverse map $\widetilde{f}^{-1}$ using a pre-computed dataset $\mathcal{D}_n = \{(v_i, p_i)\}_{i=1}^n$. The benefit of learning a model instead of using iterative optimization is that once the model is trained, prediction can be performed without the ODE system or the extensive pre-computed dataset.

Furthermore, we propose to use a kernel function $k : \mathcal{V}_T \times \mathcal{V}_T \to \mathbb{R}$ to implicitly map the AP traces into a high dimensional feature space, namely an RKHS $\mathcal{H}_k$. The features $\{\phi_i(v)\}_{i=1}^\infty$ of the RKHS give a much richer representation of the AP traces than the AP features $\{\phi_i^{AP}(v)\}_{i=1}^m$. Here $\phi_i = k(\cdot, v_i)$. Moreover, efficient comparison of AP traces is made possible by the kernel trick, $k(v_i, v_j) = \langle \phi_i, \phi_j \rangle_k$, which circumvents the need to calculate the features explicitly.

To find the best estimator in the RKHS we use kernel regularized ridge regression as this gives rise to a convex optimization problem in $\mathcal{H}_k$, thereby avoiding the issue of local minima. In the manuscript, we compare the KRR solvers StreaMRAK and FALKON, where StreaMRAK is the algorithm developed in the first paper in this thesis; see Section 6.1.

For a measured AP trace w, the performance of StreaMRAK and FALKON is compared to finding the best fit in the dataset, namely $v_{opt} = \text{argmin}_{v_i \in \mathcal{D}_n} L(v_i, w)$. For this purpose, the L_2 loss directly in $\mathcal{V}_T \subset \mathbb{R}^T$ and the L_2 loss in the AP-feature space are used. Since we are interested in comparing accuracy, it is natural to compare with v_{opt} as this is the solution that is searched for iteratively in Tveito et al. [118].

The contribution of this manuscript is to demonstrate that kernel methods are a viable modeling strategy for the problem of estimating ionic current parameters from AP trace measurements. The manuscript demonstrates that the kernel methods StreaMRAK and FALKON have significantly higher accuracy and reliability than the optimization scheme used in Tveito et al. [118]. This is important as high accuracy and reliability in predictions are essential in drug development, where errors can have severe consequences. In particular, StreaMRAK is shown to outperform FALKON both in terms of accuracy and reliability across different regions of the parameter domain.

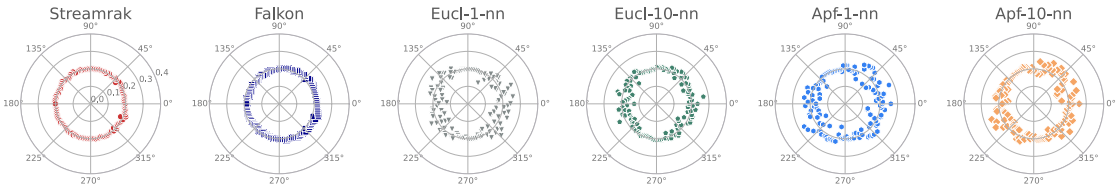


Figure 1.6: Predictions of parameters from AP traces corresponding to parameters on the circle $\mathcal{C} = \{p \in \mathcal{P} : \|p - p_0\|_2 = 0.2\}$, where $p_0 = (1, 1)$. For StreaMRAK and FALKON the predicted parameters are very close to the circle $\mathcal{C}$.

Figure 1.6 compares StreaMRAK and FALKON with four alternative parameter prediction schemes. Here Eucl-1-nn refers to $v_{opt} = \text{argmin}_{v_i \in \mathcal{D}_n} L(v_i, w)$ with the L_2 loss in $\mathcal{V}_T$ and Eucl-10-nn refers to the average over the 10 nearest neighbours as measured by this loss. Similarly, Apf-1-nn refers to v_{opt} with the L_2 loss in AP-feature space, and Apf-10-nn refers to the average over the 10 nearest neighbours. The algorithms are given AP traces corresponding to parameters sampled from the circle $\mathcal{C} = \{p \in \mathcal{P} : \|p - p_0\|_2 = 0.2\}$, where $p_0 = (1, 1)$. The goal is to predict the parameters that generated the given AP traces. From the figure, it is clear that the prediction accuracy of StreaMRAK and FALKON is higher and also more consistent in every direction in the parameter domain than the alternative schemes.

6.3 Paper III: Effective resistance in metric spaces

Effective resistance (ER) is a distance metric on graphs. An important application of this distance metric is to uncover the intrinsic structure of point clouds. However, ER suffers from a major limitation. Namely, as the graph size increases, the ER between nodes in a graph converges to a trivial limit. The latest demonstration of this problem is due to Von-Luxburg et al. [71, 125] following several other works on this issue [7, 20, 70]. This problem is commonly referred to as the Von-Luxburg limit, which we define in Proposition 11.

Proposition 11 (Von-Luxburg limit [71]) *Let $G_n = (X_n, W)$ be a graph, with nodes $X_n = \{x_1, \ldots, x_n\}$, edge weights W_{ij} and let $D_i = \sum_{j=1}^{n} W_{ij}$ be the degree of node x_i. Let $R_n(x_i, x_j)$ denote the effective resistance between node $x_i, x_j \in X_n$ defined in Proposition 8. It then follows that*

$$\lim_{n\to\infty} R_n(x_i, x_j) \propto 1/D_i + 1/D_j$$

The consequence of Proposition 11 is that in the asymptotic limit, the distance between two graph nodes $x_i, x_j \in X_n$ is only determined by their respective degrees D_i, D_j. Consequently, the ER is effectively meaningless as a distance metric. Furthermore, Von-Luxburg et al. [125] show that this problem occurs already for relatively small graphs with $n \approx 1000$ nodes.

Manuscript contribution The contribution of this manuscript is to introduce the concept of region-based ER and to demonstrate that this definition does not suffer from the trivial limit described in Proposition 11. Let $G_n = (X_n, W)$ be a graph with nodes $X_n = \{x_1, \ldots, x_n\}$ and let $X_s, X_g \subset X_n$ be two non-empty disjoint subsets. We define the region-based ER as $R_n(X_s, X_g) = 1/J_{tot}$ where

$$J_{tot} = \sum_{x_i \in X_s} \sum_{x_j \in X_n} W_{ij}(v_n^*(x_i) - v_n^*(x_j))$$

is the total current between X_s and X_g induced by the energy-minimizing voltage v_n^*. This voltage is defined as the solution to the energy minimization problem

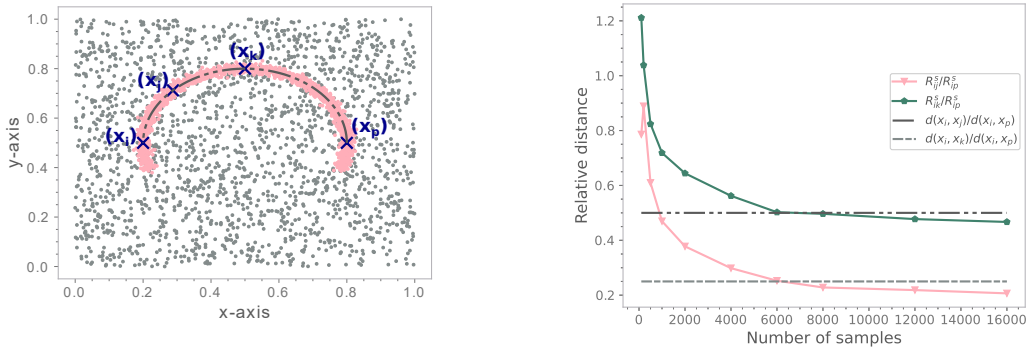
$$\begin{aligned} \min_{v: X_n \to \mathbb{R}} \quad & \sum_{x_i, x_j \in X_n} W_{i,j}(v(x_i) - v(x_j))^2 \\ \text{Subject to} \quad & v(x_i) = 1 \, \forall x_i \in X_s, \quad v(x_i) = 0, \, \forall x_i \in X_g. \end{aligned}$$

The region-based ER can be contrasted with the classical definition of ER from Proposition 8.

The region-based ER is based on the definition of ER between sets from Song et al. [110]. In the manuscript, we extend this to the setting where X_n are sampled from a distribution μ defined over some metric space (M, d). We let $X_s = \{x \in X_n : x \in M_s\}$ and $X_g = \{x \in X_n : x \in M_g\}$ where $M_s, M_g \subset M$ are disjoint measurable subsets of M.

Using a kernel function $k : M \times M \to \mathbb{R}$, combined with appropriate scaling of the edge weights, a local neighborhood graph G_n is constructed as described in Definition 6. Under certain technical conditions on M, k, and μ, the region-based ER defined on G_n is shown to converge in probability to a limit object $R_\mu(M_s, M_g)$ as $n \to \infty$. The contributions of the manuscript can be summarized as:

1. *Existence*: We prove the existence and uniqueness of $R_\mu(M_s, M_g)$
2. *Convergence*: We prove that $R_n(X_s, X_g)$ converges to $R_\mu(M_s, M_g)$ in probability as $n \to \infty$



(a) (b)

Figure 1.7: Convergence of region-based ER to a meaningful limit. (a) The pink half-moon is a high-density region over a low-density background. (b) the horizontal lines corresponds to Γ_{ijp} and Γ_{ikp}. Pink line shows R^s_{ij}/R^s_{ip} while green line shows R^s_{ik}/R^s_{ip}.

3. *Distance metric*: We prove that region-based ER is a distance metric
4. *Meaningful*: We demonstrate numerically that region-based ER converges to a meaningful limit

For a numerical example, consider the distance along the arch of the half-moon illustrated in Figure 1.7a. Let $d(\cdot,\cdot)$ denote the true distance along the arch and define the ratios

$$\Gamma_{ijp} \coloneqq d(x_i,x_j)/d(x_i,x_p) = 0.25 \quad \text{and} \quad \Gamma_{ikp} \coloneqq d(x_i,x_k)/d(x_i,x_p) = 0.5 \quad (1.18)$$

Let $R_{ij} \coloneqq R_n(X_i,X_j)$. As seen from Figure 1.7, when n increases, the ratios of the region-based ER, namely R_{ij}/R_{ip} and R_{ik}/R_{ip}, converges to values close to the ratios Γ_{ijp} and Γ_{ikp} respectively. Note that since the ER incorporates all possible paths between the two nodes, we do not expect the region-based ER to converge exactly to Γ_{ijp} and Γ_{ikp}. However, since the density of the half-moon is significantly higher than that of the background, we expect the limits to be close. On the other hand, with the limit in Proposition 11, the ratios are 1. This is because the density on the half-moon is uniform, which means the respective degrees of the nodes are the same.

6.4 Paper IV: Structure from voltage

Non-linear dimensionality reduction (NLDR) is the discipline of finding lower-dimensional representations of non-linear data to reduce the impact of the curse of dimensionality. As datasets are rapidly growing in size, developing scalable NLDR algorithms is becoming increasingly important.

A powerful strategy to achieve scalability is to utilize powerful computational models such as parallelization, distribution, and streaming. However, existing NLDR techniques based on eigenfunction calculations, such as Laplacian eigenmaps [14], are generally incompatible with these computational models. From

the discussion in Section 4.2, we know that part of the problem with LE is that it does not provide guarantees for the functions to be localized. Whereby localized, we mean in the sense defined in Definition 7. Furthermore, demanding the eigenfunctions to be orthogonal means they can not be computed independently. Moreover, once the eigenfunctions are calculated, they can not be extended to new samples without repeating the process.

Manuscript contributions In this manuscript, we propose a novel embedding scheme based on localized voltage functions that can be calculated independently, thereby allowing them to be computed using parallelization and distributed schemes. The voltage functions we define can also be easily extended to new samples, which makes them compatible with a streaming model of computation. We refer to these voltage functions as *grounded metric voltage* functions (GMVs) denoted v_{n,s_i}. The proposed embedding is

$$x_i \mapsto (v_{n,s_1}(x_i), \dots v_{n,s_m}(x_i))^\top. \tag{1.19}$$

Consider a setting where $X_n = \{x_1, \dots, x_n\}$ is sampled from a distribution μ over some metric space (M, d). Let $k : M \times M \to [0, 1]$ be a kernel function and let (X_n, W) be a graph with edge weights $W_{ij} = k(x_i, x_j)/n^2$. Furthermore, let r_s be some radius and $g : \mathbb{R} \to [0, r]$ with $r > 0$ be a monotonic strictly decreasing function. Define $X_s = \{x \in X_n : x \in M_s\}$, where $M_s = \{x \in M : g(d(x, x_s)) \leq r_s\}$. Here $x_s \in M$ is what we call a source center. For a given source x_s, we define the associated GMV function $v_{n,s} : X_n \to [0, 1]$ as the solution to the energy minimization problem

$$\begin{aligned} &\min_{v:X\to[0,1]} \sum_{x_i,x_j\in X_n} W_{ij}(v(x_i) - v(x_j))^2 + \sum_{x_i\in X_n} \rho v^2(x_i) \\ &\text{Subject to} \quad v(x_i) = 1 \quad \text{for all} \quad x_i \in X_s. \end{aligned}$$

The term $\sum_{x_i} \rho v^2(x_i)$ incorporates the effect of an universal ground $x_g \notin X_n$, with voltage $v(x_g) = 0$, that connects to all nodes in X_n with edge weight $\rho = \rho_g/n$ for $\rho_g > 0$. This can easily be seen by considering $X_n \cup \{x_g\}$ and adding an extra row and column to W_{ij} with the weight ρ. The term $\sum_{x_i\in X_n\cup\{x_g\}} \rho(v(x_i) - v(x_g))^2 = \sum_{x_i\in X_n\cup\{x_g\}} \rho v^2(x_i)$ can then be extracted from the sum. Since the voltage of the ground is anyway $v(x_g) = 0$, we drop the sum over x_g and ignore the ground in constructing W_{ij}.

We note that in a random walk perspective, the ground can be interpreted as a trap node with zero escape probability.

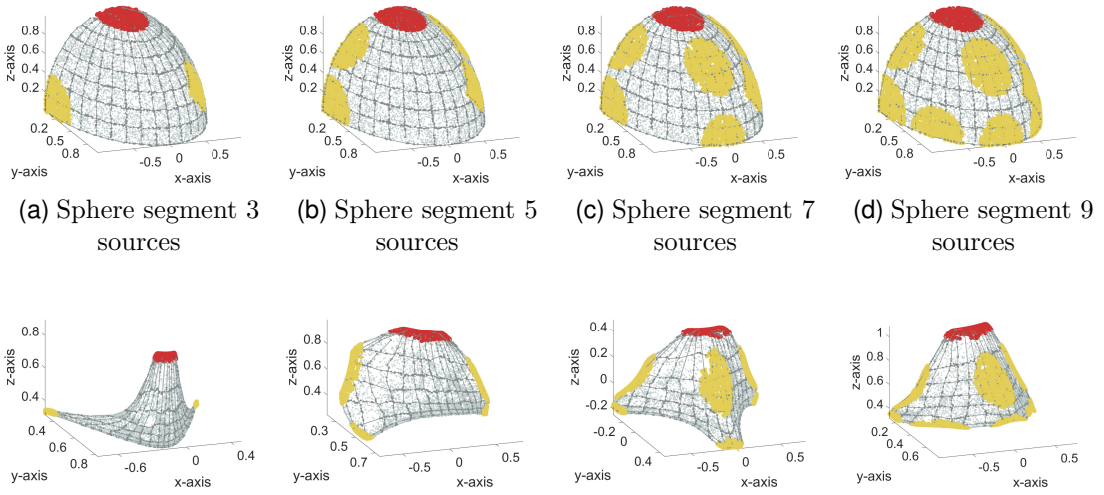
The idea of the source and ground constraints is that together, they create a voltage function localized around x_s. The contributions of the manuscript can be summarized as follows:

1. *Existence*: We prove the existence and uniqueness of a limit object v_s^*.

2. *Convergence*: We prove that $v_{n,s}$ converges to v_s^* in probability as $n \to \infty$.

3. *Locality*: We provide bounds on the shape of v_s^* on the unit sphere S^{d-1}, proving that $v_s^*(x_i)$ decays exponentially with increasing $d_S(x_i, x_s)$, with decay governed by the magnitude of ρ. Here d_S is the geodesic on S^{d-1}.

4. *Embedding*: We show analytically and numerically how the GMV can provide an embedding of the unit sphere.



(a) Sphere segment 3 sources (b) Sphere segment 5 sources (c) Sphere segment 7 sources (d) Sphere segment 9 sources

(e) Embedding 3 sources (f) Embedding 5 sources (g) Embedding 7 sources (h) Embedding 9 sources

Figure 1.8: Embedding of the two first quadrants of the unit sphere. The yellow and red circles are source regions. The upper row shows the two first quadrants of the unit sphere, and the lower row is the embedding $x_i \mapsto (v_{n,s_1}(x_i), \ldots, v_{n,s_m}(x_i))$ for $m \in \{3, 5, 7, 9\}$.

It can be shown that v_{n,s_i} satisfies

$$v_{n,s_i} = \widetilde{D}^{-1}\widetilde{W}^{(s_i)}v_{n,s_i}$$

where $\widetilde{W}^{(s_i)}$ is referred to as the grounded weight matrix. This means that v_{n,s_i} can be found by power iteration where $\widetilde{D}^{-1}\widetilde{W}^{(s_i)}$ is applied iteratively until convergence. Due to the locality of v_{n,s_i}, it follows that only a sub-matrix is necessary, which greatly reduces the computational expense of the iterative procedure.

In Figure 1.8, an embedding $x_i \mapsto (v_{n,s_1}(x_i), \ldots, v_{n,s_m}(x_i))$ of the 2-dimensional unit sphere embedded in $\mathbb{R}^D$ is demonstrated for $m \in \{3, 5, 7, 9\}$. Here $m \ll D$. For visualization, multidimensional scaling [35] is used to project the representation into $\mathbb{R}^3$.

7 Summary and outlook

Learning algorithms based on a kernel function are theoretically well understood in statistical learning theory and machine learning, which makes them attractive learning algorithms in terms of reliability and interpretability. However, in

their basic form, they suffer from large memory requirements and computational costs, preventing their utility in real-world applications where datasets are huge. Therefore, to allow the extension of these algorithms to big data, the development of scalable kernel-based learning schemes is of interest.

In this thesis, we made several contributions that improve the scalability of kernel-based learning. The novel KRR solver StreaMRAK developed in paper I and further demonstrated in paper II, has shown promise as an efficient KRR solver with improvements over the existing KRR solver FALKON.

Future work should focus on studying the generalization properties of StreaMRAK. In particular, in the asymptotic limit, FALKON has estimation rates that are optimal in a min-max sense [95]. As such StreaMRAK can not achieve better in this limit. However, our numerical experiments suggest that StreaMRAK can reach high predictive accuracy with significantly fewer samples than FALKON. The characterization of the generalization properties of StreaMRAK is therefore of great interest to gain insights on how the boosting and adaptive sub-sampling impact the learning.

In the second paper of this thesis, StreaMRAK was demonstrated as a reliable algorithm for estimating ionic membrane currents from cardiac AP traces. This is an important problem within cardiac anti-arrhythmic research and cardiac drug development. Therefore, a further demonstration of StreaMRAK in this field is of great interest. In particular, the important aspects of AP trace inversion are scalability, reliability, and the ability to handle parameters with identifiability and sensitivity issues. The study in paper II focuses on demonstrating StreaMRAK in terms of its reliability and its ability to detect low-sensitivity parameters. The study is performed in a controlled setting with a limited number of parameters. Future work should focus on identifiability issues and, finally, on applying StreaMRAK to modern AP models with hundreds of parameters.

The third paper in this thesis has demonstrated how the region-based ER avoids the convergence issues of standard ER. Future work should focus on characterizing the dependency between source radius, kernel bandwidth, the weight-to-ground, and graph size. The consequences of decreasing the source radius are of particular interest.

The theoretical and numerical results from paper IV lay the foundations for an embedding scheme utilizing grounded metric voltage functions. For future work, there are in particular two directions of interest. We are currently working towards extending the embedding scheme to more general manifolds and real-world data. Furthermore, the computational and algorithmic aspects of the embedding scheme require further development. In particular, our results show that the GMVs are localized and can be computed independently in an iterative manner, indicating compatibility with distribution and streaming. Current efforts are therefore dedicated to algorithmic developments that can utilize these properties.

Bibliography

[1] Ittai Abraham, Yair Bartal and Ofer Neimany. 'Advances in metric embedding theory'. In: *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. 2006, pp. 271–286. DOI: 10.1145/1132516.1132557.

[2] MA Aiserman, Emmanuil M Braverman and Lev I Rozonoer. 'Theoretical foundations of the potential function method in pattern recognition'. In: *Avtomat. i Telemeh.* 25.6 (1964), pp. 917–936.

[3] Ahmed Alaoui and Michael W Mahoney. 'Fast randomized kernel ridge regression with statistical guarantees'. In: *Advances in neural information processing systems* 28 (2015). URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/f3f27a324736617f20abbf2ffd806f6d-Paper.pdf.

[4] William K Allard, Guangliang Chen and Mauro Maggioni. 'Multi-scale geometric methods for data sets II: Geometric multi-resolution analysis'. In: *Applied and computational harmonic analysis* 32.3 (2012), pp. 435–462. DOI: 10.1016/j.acha.2011.08.001.

[5] Elena A Allen, Erik B Erhardt and Vince D Calhoun. 'Data visualization in the neurosciences: overcoming the curse of dimensionality'. In: *Neuron* 74.4 (2012), pp. 603–608. DOI: 10.1016/j.neuron.2012.05.001.

[6] Sylvain Arlot and Alain Celisse. 'A survey of cross-validation procedures for model selection'. In: *Statistics Surveys* 4 (2010), pp. 40–79. DOI: 10.1214/09-SS054.

[7] Chen Avin and Gunes Ercal. 'On the cover time and mixing time of random geometric graphs'. In: *Theoretical Computer Science* 380.1-2 (2007), pp. 2–22. DOI: 10.1016/j.tcs.2007.02.065.

[8] Ariful Azad and Aydin Buluç. 'A Work-Efficient Parallel Sparse Matrix-Sparse Vector Multiplication Algorithm'. In: *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. 2017, pp. 688–697. DOI: 10.1109/IPDPS.2017.76.

[9] Francis Bach. 'Sharp analysis of low-rank kernel matrix approximations'. In: *Proceedings of the 26th Annual Conference on Learning Theory*. Vol. 30. Proceedings of Machine Learning Research. Princeton, NJ, USA, 2013, pp. 185–209. URL: https://proceedings.mlr.press/v30/Bach13.html.

[10] Maroua Bahri et al. 'Data stream analysis: Foundations, major tasks and tools'. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 11.3 (2021), e1405. DOI: 10.1002/widm.1405.

[11] Ron Bekkerman, Mikhail Bilenko and John Langford. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

[12] Mikhail Belkin. 'Problems of learning on manifolds'. PhD thesis. The University of Chicago, 2004.

[13] Mikhail Belkin and Partha Niyogi. 'Convergence of Laplacian Eigenmaps'. In: *Advances in Neural Information Processing Systems*. Vol. 19. 2006. URL: https://proceedings.neurips.cc/paper_files/paper/2006/file/5848ad959570f87753a60ce8be1567f3-Paper.pdf.

[14] Mikhail Belkin and Partha Niyogi. 'Laplacian Eigenmaps for Dimensionality Reduction and Data Representation'. In: *Neural Computation* 15.6 (2003), pp. 1373–1396. DOI: 10.1162/089976603321780317.

[15] Mikhail Belkin and Partha Niyogi. 'Semi-supervised learning on manifolds'. In: *Machine Learning Journal* 1 (2002).

[16] Mikhail Belkin and Partha Niyogi. 'Towards a theoretical foundation for Laplacian-based manifold methods'. In: *Journal of Computer and System Sciences* 74.8 (2008), pp. 1289–1308. DOI: 10.1016/j.jcss.2007.08.006.

[17] R. Bellman and R. Kalaba. 'On adaptive control processes'. In: *IRE Transactions on Automatic Control* 4.2 (1959), pp. 1–9. DOI: 10.1109/TAC.1959.1104847.

[18] Visar Berisha et al. 'Digital medicine and the curse of dimensionality'. In: *NPJ digital medicine* 4.1 (2021), p. 153. URL: https://doi.org/10.1038/s41746-021-00521-5.

[19] Bernhard E Boser, Isabelle M Guyon and Vladimir N Vapnik. 'A training algorithm for optimal margin classifiers'. In: *Proceedings of the fifth annual workshop on Computational learning theory*. 1992, pp. 144–152. DOI: 10.1145/130385.130401.

[20] Stephen P Boyd et al. 'Mixing Times for Random Walks on Geometric Random Graphs.' In: *ALENEX/ANALCO*. 2005, pp. 240–249.

[21] Sergey Brin and Lawrence Page. 'The anatomy of a large-scale hypertextual Web search engine'. In: *Computer Networks and ISDN Systems* 30.1 (1998), pp. 107–117. DOI: 10.1016/S0169-7552(98)00110-X.

[22] Fred Buckley and Frank Harary. *Distance in graphs*. Vol. 2. Addison-Wesley Redwood City, 1990.

[23] Peter J. Burt and Edward H. Adelson. 'The Laplacian Pyramid as a Compact Image Code'. In: *Readings in Computer Vision*. Morgan Kaufmann, 1987, pp. 671–679. DOI: 10.1016/B978-0-08-051581-6.50065-9.

[24] Raffaello Camoriano et al. 'NYTRO: When Subsampling Meets Early Stopping'. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*. Vol. 51. Proceedings of Machine Learning Research. 2016, pp. 1403–1411. URL: https://proceedings.mlr.press/v51/camoriano16.html.

[25] Gavin C. Cawley and Nicola L.C. Talbot. 'Fast exact leave-one-out cross-validation of sparse least-squares support vector machines'. In: *Neural Networks* 17.10 (2004), pp. 1467–1475. DOI: 10.1016/j.neunet.2004.07.002.

[26] Chris Chambers et al. 'High-throughput screening of Na_V1. 7 modulators using a giga-seal automated patch clamp instrument'. In: *Assay and drug development technologies* 14.2 (2016), pp. 93–108. DOI: 10.1089/adt.2016.700.

[27] Fulong Chen et al. 'Identification of the Parameters of the Beeler–Reuter Ionic Equation With a Partially Perturbed Particle Swarm Optimization'. In: *IEEE Transactions on Biomedical Engineering* 59.12 (2012), pp. 3412–3421. DOI: 10.1109/TBME.2012.2216265.

[28] Mike Clements and Nick Thomas. 'High-Throughput Multi-Parameter Profiling of Electrophysiological Drug Effects in Human Embryonic Stem Cell Derived Cardiomyocytes Using Multi-Electrode Arrays'. In: *Toxicological Sciences* 140.2 (2014), pp. 445–461. DOI: 10.1093/toxsci/kfu084.

[29] Michael B Cohen et al. 'Uniform sampling for matrix approximation'. In: *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*. 2015, pp. 181–190. DOI: 10.1145/2688073.2688113.

[30] Ronald R Coifman and Stéphane Lafon. 'Diffusion maps'. In: *Applied and computational harmonic analysis* 21.1 (2006), pp. 5–30. DOI: 10.1016/j.acha.2006.04.006.

[31] Ronald R Coifman et al. 'Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps'. In: *Proceedings of the national academy of sciences* 102.21 (2005), pp. 7426–7431. DOI: 10.1073/pnas.0500334102.

[32] Corinna Cortes and Vladimir Vapnik. 'Support-vector networks'. In: *Machine learning* 20 (1995), pp. 273–297. DOI: 10.1007/BF00994018.

[33] Jose A Costa and Alfred O Hero. 'Learning intrinsic dimension and intrinsic entropy of high-dimensional datasets'. In: *2004 12th European Signal Processing Conference*. IEEE. 2004, pp. 369–372.

[34] Felipe Cucker and Steve Smale. 'On the mathematical foundations of learning'. In: *Bulletin of the American mathematical society* 39.1 (2002), pp. 1–49.

[35] Ivan Dokmanic et al. 'Euclidean Distance Matrices: Essential theory, algorithms, and applications'. In: *IEEE Signal Processing Magazine* 32.6 (2015), pp. 12–30. DOI: 10.1109/MSP.2015.2398954.

[36] Petros Drineas, Ravi Kannan and Michael W Mahoney. 'Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication'. In: *SIAM Journal on Computing* 36.1 (2006), pp. 132–157. DOI: 10.1137/S0097539704442684.

[37] Petros Drineas, Ravi Kannan and Michael W Mahoney. 'Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix'. In: *SIAM Journal on computing* 36.1 (2006), pp. 158–183. DOI: 10.1137/S0097539704442696.

[38] Petros Drineas et al. 'Fast approximation of matrix coherence and statistical leverage'. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 3475–3506.

[39] Andrew G Edwards and William E Louch. 'Species-dependent mechanisms of cardiac arrhythmia: a cellular focus'. In: *Clinical Medicine Insights: Cardiology* 11 (2017), pp. 1179–5468. DOI: 10.1177/1179546816686061.

[40] Artur J Ferreira and Mário AT Figueiredo. 'Boosting algorithms: A review of methods, theory, and applications'. In: *Ensemble machine learning: Methods and applications* (2012), pp. 35–85.

[41] Shai Fine and Katya Scheinberg. 'Efficient SVM training using low-rank kernel representations'. In: *Journal of Machine Learning Research* 2.Dec (2001), pp. 243–264.

[42] Piero Colli Franzone, Luca Franco Pavarino and Simone Scacchi. *Mathematical cardiac electrophysiology*. Vol. 13. Springer, 2014.

[43] Yoav Freund. 'Boosting a weak learning algorithm by majority'. In: *Information and computation* 121.2 (1995), pp. 256–285. DOI: 10.1006/inco.1995.1136.

[44] Yoav Freund and Robert E Schapire. 'A decision-theoretic generalization of on-line learning and an application to boosting'. In: *Journal of computer and system sciences* 55.1 (1997), pp. 119–139. DOI: 10.1006/jcss.1997.1504.

[45] Jerome H. Friedman. 'Greedy function approximation: A gradient boosting machine.' In: *The Annals of Statistics* 29.5 (2001), pp. 1189–1232. DOI: 10.1214/aos/1013203451.

[46] Jerome H. Friedman. 'Stochastic gradient boosting'. In: *Computational Statistics & Data Analysis* 38.4 (2002), pp. 367–378. DOI: 10.1016/S0167-9473(01)00065-2.

[47] Heitor Murilo Gomes et al. 'Machine learning for streaming data: state of the art, challenges, and opportunities'. In: *ACM SIGKDD Explorations Newsletter* 21.2 (2019), pp. 6–22. DOI: 10.1145/3373464.3373470.

[48] Eleonora Grandi, Francesco S. Pasqualini and Donald M. Bers. 'A novel computational model of the human ventricular action potential and Ca transient'. In: *Journal of Molecular and Cellular Cardiology* 48.1 (2010), pp. 112–121. DOI: 10.1016/j.yjmcc.2009.09.019.

[49] László Györfi et al. *A distribution-free theory of nonparametric regression*. Vol. 1. Springer, 2002.

[50] Jihun Ham et al. 'A kernel view of the dimensionality reduction of manifolds'. In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 47. DOI: 10.1145/1015330.1015417.

[51] Trevor Hastie et al. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[52] Matthias Hein, Jean-Yves Audibert and Ulrike von Luxburg. 'Graph Laplacians and their convergence on random neighborhood graphs.' In: *Journal of Machine Learning Research* 8.6 (2007).

[53] Todd J Herron, Peter Lee and José Jalife. 'Optical imaging of voltage and calcium in cardiac cells & tissues'. In: *Circulation research* 110.4 (2012), pp. 609–623. DOI: 10.1161/CIRCRESAHA.111.247494.

[54] Thomas Hofmann, Bernhard Schölkopf and Alexander J. Smola. 'Kernel methods in machine learning'. In: *The Annals of Statistics* 36.3 (2008), pp. 1171–1220. DOI: 10.1214/009053607000000677.

[55] Alan Julian Izenman. 'Introduction to manifold learning'. In: *Wiley Interdisciplinary Reviews: Computational Statistics* 4.5 (2012), pp. 439–446. DOI: 10.1002/wics.1222.

[56] Karoline Horgmo Jæger et al. 'Identifying Drug Response by Combining Measurements of the Membrane Potential, the Cytosolic Calcium Concentration, and the Extracellular Potential in Microphysiological Systems'. In: *Frontiers in Pharmacology* 11 (2021), pp. 569–489. DOI: https://doi.org/10.3389/fphar.2020.569489.

[57] Karoline Horgmo Jæger et al. 'Improved Computational Identification of Drug Response Using Optical Measurements of Human Stem Cell Derived Cardiomyocytes in Microphysiological Systems'. In: *Frontiers in Pharmacology* 10 (2020). DOI: 10.3389/fphar.2019.01648.

[58] Palle ET Jorgensen and PJ Pearse Erin. 'Operator theory and analysis of infinite networks'. In: *arXiv preprint arXiv:0806.3881* 3 (2008).

[59] George Kimeldorf and Grace Wahba. 'Some results on Tchebycheffian spline functions'. In: *Journal of mathematical analysis and applications* 33.1 (1971), pp. 82–95.

[60] George S. Kimeldorf and Grace Wahba. 'A Correspondence Between Bayesian Estimation on Stochastic Processes and Smoothing by Splines'. In: *Ann. Math. Stat.* 41.2 (2011), pp. 495–502. DOI: 10.1214/aoms/1177697089.

[61] Douglas J Klein and Milan Randić. 'Resistance distance'. In: *Journal of mathematical chemistry* 12.1 (1993), pp. 81–95. DOI: 10.1007/BF01164627.

[62] Bruce G. Kornreich. 'The patch clamp technique: Principles and technical considerations'. In: *Journal of Veterinary Cardiology* 9.1 (2007), pp. 25–37. DOI: 10.1016/j.jvc.2007.02.001.

[63] John A Lee, Michel Verleysen et al. *Nonlinear dimensionality reduction*. Vol. 1. Springer, 2007.

[64] Wenjing Liao and Mauro Maggioni. 'Adaptive Geometric Multiscale Approximations for Intrinsically Low-dimensional Data.' In: *Journal of Machine Learning Research* 20 (2019), pp. 98–1.

[65] Wenjing Liao, Mauro Maggioni and Stefano Vigogna. 'Learning adaptive multiscale approximations to data and functions near low-dimensional sets'. In: *IEEE Information Theory Workshop*. IEEE. 2016, pp. 226–230. DOI: 10.1109/ITW.2016.7606829.

[66] Shao-Bo Lin, Yunwen Lei and Ding-Xuan Zhou. 'Boosted Kernel Ridge Regression: Optimal Learning Rates and Early Stopping'. In: *Journal of Machine Learning Research* 20.46 (2019), pp. 1–36. URL: http://jmlr.org/papers/v20/18-063.html.

[67] Anna V Little, Mauro Maggioni and Lorenzo Rosasco. 'Multiscale geometric methods for data sets I: Multiscale SVD, noise and curvature'. In: *Applied and Computational Harmonic Analysis* 43.3 (2017), pp. 504–567. DOI: 10.1016/j.acha.2015.09.009.

[68] Axel Loewe et al. 'Parameter Estimation of Ion Current Formulations Requires Hybrid Optimization Approach to Be Both Accurate and Reliable'. In: *Frontiers in Bioengineering and Biotechnology* 3 (2016). DOI: 10.3389/fbioe.2015.00209.

[69] Wei-Yin Loh. 'Classification and regression trees'. In: *Wiley interdisciplinary reviews: data mining and knowledge discovery* 1.1 (2011), pp. 14–23. DOI: https://doi.org/10.1002/widm.8.

[70] László Lovász. 'Random walks on graphs: A survey'. In: *Combinatorics* 2 (1993), pp. 1–46.

[71] Ulrike Luxburg, Agnes Radl and Matthias Hein. 'Getting lost in space: Large sample analysis of the resistance distance'. In: *Advances in Neural Information Processing Systems*. Vol. 23. Curran Associates, Inc., 2010. URL: https://proceedings.neurips.cc/paper/2010/file/0d0871f0806eae32d30983b62252da50-Paper.pdf.

[72] Mauro Maggioni, Stanislav Minsker and Nate Strawn. 'Multiscale Dictionary Learning: Non-Asymptotic Bounds and Robustness'. In: *Journal of Machine Learning Research* 17.2 (2016), pp. 1–51. URL: http://jmlr.org/papers/v17/maggioni16a.html.

[73] Michael W Mahoney and Petros Drineas. 'CUR matrix decompositions for improved data analysis'. In: *Proceedings of the National Academy of Sciences* 106.3 (2009), pp. 697–702.

[74] Michael W. Mahoney. 'Randomized Algorithms for Matrices and Data'. In: *Foundations and Trends in Machine Learning* 3.2 (2011), pp. 123–224. DOI: 10.1561/2200000035.

[75] Abdul Majeed and Ibtisam Rauf. 'Graph theory: A comprehensive survey about graph theory applications in computer science and social networks'. In: *Inventions* 5.1 (2020), p. 10. DOI: 10.3390/inventions5010010.

[76] Per-Gunnar Martinsson and Joel A Tropp. 'Randomized numerical linear algebra: Foundations and algorithms'. In: *Acta Numerica* 29 (2020), pp. 403–572. DOI: 10.1017/S0962492920000021.

[77] Anurag Mathur et al. 'Human iPSC-based Cardiac Microphysiological System For Drug Screening Applications'. In: *Scientific Reports* 5.1 (Mar. 2015). DOI: https://doi.org/10.1038/srep08883.

[78] Giacomo Meanti et al. 'Kernel methods through the roof: handling billions of points efficiently'. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 14410–14422. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/a59afb1b7d82ec353921a55c579ee26d-Paper.pdf.

[79] Charles A. Micchelli, Yuesheng Xu and Haizhang Zhang. 'Universal Kernels'. In: *Journal of Machine Learning Research* 7.95 (2006), pp. 2651–2667. URL: http://jmlr.org/papers/v7/micchelli06a.html.

[80] Berndt Müller, Joachim Reinhardt and Michael T Strickland. *Neural networks: an introduction*. Springer Science & Business Media, 1995.

[81] Shanmugavelayutham Muthukrishnan et al. 'Data streams: Algorithms and applications'. In: *Foundations and Trends in Theoretical Computer Science* 1.2 (2005), pp. 117–236. DOI: http://dx.doi.org/10.1561/0400000002.

[82] Erich Novak and Hans Triebel. 'Function Spaces in Lipschitz Domains and Optimal Rates of Convergence for Sampling.' In: *Constructive approximation* 23.3 (2006).

[83] Thomas O'Hara et al. 'Simulation of the Undiseased Human Cardiac Ventricular Action Potential: Model Formulation and Experimental Validation'. In: *PLOS Computational Biology* 7.5 (May 2011), pp. 1–29. DOI: https://doi.org/10.1371/journal.pcbi.1002061.

[84] Phillip A Ostrand. 'Covering dimension in general spaces'. In: *General Topology and its applications* 1.3 (1971), pp. 209–221. DOI: https://doi.org/10.1016/0016-660X(71)90093-6.

[85] Lawrence Page et al. *The PageRank citation ranking: Bringing order to the web.* Tech. rep. Stanford infolab, 1999.

[86] Georgios A Pavlopoulos et al. 'Using graph theory to analyze biological networks'. In: *BioData mining* 4 (2011), pp. 1–27. DOI: https://doi.org/10.1186/1756-0381-4-10.

[87] Zhilin Qu et al. 'Nonlinear and stochastic dynamics in the heart'. In: *Physics Reports* 543.2 (2014), pp. 61–162. DOI: https://doi.org/10.1016/j.physrep.2014.05.002.

[88] Zhilin Qu et al. 'Nonlinear and stochastic dynamics in the heart'. In: *Physics Reports* 543.2 (2014), pp. 61–162. DOI: https://doi.org/10.1016/j.physrep.2014.05.002.

[89] Alfio Quarteroni et al. 'Integrated Heart—Coupling multiscale and multiphysics models for the simulation of the cardiac function'. In: *Computer Methods in Applied Mechanics and Engineering* 314 (2017), pp. 345–407. DOI: https://doi.org/10.1016/j.cma.2016.05.031.

[90] Ali Rahimi and Benjamin Recht. 'Random features for large-scale kernel machines'. In: *Advances in neural information processing systems* 20 (2007). URL: https://proceedings.neurips.cc/paper_files/paper/2007/file/013a006f03dbc5392effeb8f18fda755-Paper.pdf.

[91] Ferozuddin Riaz and Khidir M Ali. 'Applications of graph theory in computer science'. In: *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*. IEEE. 2011, pp. 142–145. DOI: 10.1109/CICSyN.2011.40.

[92] Lorenzo Rosasco, Mikhail Belkin and Ernesto De Vito. 'On Learning with Integral Operators'. In: *Journal of Machine Learning Research* 11.30 (2010), pp. 905–934. URL: http://jmlr.org/papers/v11/rosasco10a.html.

[93] Sam T. Roweis and Lawrence K. Saul. 'Nonlinear Dimensionality Reduction by Locally Linear Embedding'. In: *Science* 290.5500 (2000), pp. 2323–2326. DOI: 10.1126/science.290.5500.2323.

[94] Alessandro Rudi, Raffaello Camoriano and Lorenzo Rosasco. 'Less is more: Nyström computational regularization'. In: *Advances in Neural Information Processing Systems* 28 (2015). URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/03e0704b5690a2dee1861dc3ad3316c9-Paper.pdf.

[95] Alessandro Rudi, Luigi Carratino and Lorenzo Rosasco. 'Falkon: An optimal large scale kernel method'. In: *Advances in neural information processing systems* 30 (2017). URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/05546b0e38ab9175cd905eebcc6ebb76-Paper.pdf.

[96] Alessandro Rudi et al. 'On fast leverage score sampling and optimal learning'. In: *Advances in Neural Information Processing Systems* 31 (2018). URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/56584778d5a8ab88d6393cc4cd11e090-Paper.pdf.

[97] Y. Rudy. 'From Genes and Molecules to Organs and Organisms: Heart'. In: *Comprehensive Biophysics*. Elsevier, 2012, pp. 268–327. DOI: https://doi.org/10.1016/B978-0-12-374920-8.00924-3.

[98] Yoram Rudy and Jonathan R. Silva. 'Computational biology in the study of cardiac ion channels and cell electrophysiology'. In: *Quarterly Reviews of Biophysics* 39.1 (2006), pp. 57–116. DOI: https://doi.org/10.1017/S0033583506004227.

[99] Craig Saunders, Alexander Gammerman and Volodya Vovk. 'Ridge regression learning algorithm in dual variables'. In: *Proceedings of the 15-th International Conference on Machine Learning*. 1998, pp. 515–521.

[100] Robert E Schapire. 'The strength of weak learnability'. In: *Machine learning* 5 (1990), pp. 197–227. DOI: https://doi.org/10.1007/BF00116037.

[101] Bernhard Schölkopf, Ralf Herbrich and Alex J. Smola. 'A generalized representer theorem'. In: *Int. Conf. Comput. Learn. Theory*. 2001, pp. 416–426. DOI: https://doi.org/10.1007/3-540-44581-1_27.

[102] Bernhard Schölkopf, Alexander Smola and Klaus-Robert Müller. 'Kernel principal component analysis'. In: *International conference on artificial neural networks*. Springer. 1997, pp. 583–588. DOI: https://doi.org/10.1007/BFb0020217.

[103] Bernhard Schölkopf, Alexander Smola and Klaus-Robert Müller. 'Nonlinear component analysis as a kernel eigenvalue problem'. In: *Neural computation* 10.5 (1998), pp. 1299–1319. DOI: 10.1162/089976698300017467.

[104] Bernhard Schölkopf, Alexander J Smola, Francis Bach et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[105] John Shawe-Taylor, Nello Cristianini et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[106] Yanning Shen, Tianyi Chen and Georgios B Giannakis. 'Random feature-based online multi-kernel learning in environments with unknown dynamics'. In: *The Journal of Machine Learning Research* 20.1 (2019), pp. 773–808.

[107] Si Si, Cho-Jui Hsieh and Inderjit Dhillon. 'Memory efficient kernel approximation'. In: *International Conference on Machine Learning*. PMLR. 2014, pp. 701–709.

[108] Vikas Sindhwani, Partha Niyogi and Mikhail Belkin. 'Beyond the point cloud: from transductive to semi-supervised learning'. In: *Proceedings of the 22nd international conference on Machine learning*. 2005, pp. 824–831. DOI: https://doi.org/10.1145/1102351.1102455.

[109] Alexander J Smola. 'Sparse greedy matrix approximation for machine learning'. In: *Proceedings of the 17th international conference on machine learning, June 29-July 2 2000*. Morgan Kaufmann. 2000.

[110] Yue Song, David J Hill and Tao Liu. 'On extension of effective resistance with application to graph laplacian definiteness and power network stability'. In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 66.11 (2019), pp. 4415–4428. DOI: 10.1109/TCSI.2019.2929180.

[111] Daniel Spielman. 'Spectral graph theory'. In: *Combinatorial scientific computing* 18 (2012).

[112] Daniel A Spielman and Shang-Hua Teng. 'Spectral partitioning works: Planar graphs and finite element meshes'. In: *Linear Algebra and its Applications* 421.2-3 (2007), pp. 284–305. DOI: https://doi.org/10.1016/j.laa.2006.07.020.

[113] Vivienne Sze et al. 'Efficient processing of deep neural networks: A tutorial and survey'. In: *Proceedings of the IEEE* 105.12 (2017), pp. 2295–2329. DOI: 10.1109/JPROC.2017.2761740.

[114] Kirsten HWJ Ten Tusscher and Alexander V Panfilov. 'Alternans and spiral breakup in a human ventricular tissue model'. In: *American Journal of Physiology-Heart and Circulatory Physiology* 291.3 (2006), H1088–H1100. DOI: https://doi.org/10.1177/2168479018795117.

[115] Joshua B. Tenenbaum, Vin de Silva and John C. Langford. 'A Global Geometric Framework for Nonlinear Dimensionality Reduction'. In: *Science* 290.5500 (2000), pp. 2319–2323. DOI: 10.1126/science.290.5500.2319.

[116] Reginald P Tewarson and Reginald P Tewarson. *Sparse matrices*. Vol. 69. Academic press New York, 1973.

[117] Aslak Tveito et al. 'A Cell-Based Framework for Numerical Modeling of Electrical Conduction in Cardiac Tissue'. In: *Frontiers in Physics* 5 (2017). DOI: https://doi.org/10.3389/fphy.2017.00048.

[118] Aslak Tveito et al. 'Inversion and computational maturation of drug response using human stem cell derived cardiomyocytes in microphysiological systems'. In: *Scientific reports* 8.1 (2018), pp. 1–14. DOI: https://doi.org/10.1038/s41598-018-35858-7.

[119] Stephen Tyree et al. 'Parallel boosted regression trees for web search ranking'. In: *Proceedings of the 20th international conference on World wide web*. 2011, pp. 387–396. DOI: https://doi.org/10.1145/1963405.1963461.

[120] Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik et al. 'Dimensionality reduction: a comparative Review'. In: *Journal of Machine Learning Research* 10.66-71 (2009), p. 13.

[121] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

[122] Michel Verleysen and Damien François. 'The Curse of Dimensionality in Data Mining and Time Series Prediction'. In: *Computational Intelligence and Bioinspired Systems*. 2005, pp. 758–770. DOI: https://doi.org/10.1007/11494669_93.

[123] Nakul Verma, Samory Kpotufe and Sanjoy Dasgupta. 'Which spatial partition trees are adaptive to intrinsic dimension?' In: *arXiv preprint arXiv:1205.2609* (2012).

[124] Ulrike Von Luxburg. 'A tutorial on spectral clustering'. In: *Statistics and computing* 17 (2007), pp. 395–416. DOI: https://doi.org/10.1007/s11222-007-9033-z.

[125] Ulrike Von Luxburg, Agnes Radl and Matthias Hein. 'Hitting and commute times in large random neighborhood graphs'. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1751–1798.

[126] Grace Wahba. *Spline models for observational data*. SIAM, 1990.

[127] Douglas Brent West et al. *Introduction to graph theory*. Vol. 2. Prentice hall Upper Saddle River, 2001.

[128] Christopher Williams and Matthias Seeger. 'Using the Nyström method to speed up kernel machines'. In: *Advances in neural information processing systems* 13 (2000). URL: https://proceedings.neurips.cc/paper_files/paper/2000/file/19de10adbaa1b2ee13f77f679fa1483a-Paper.pdf.

[129] Elizabeth A Woodcock and Scot J Matkovich. 'Cardiomyocytes structure, function and associated pathologies'. In: *The international journal of biochemistry & cell biology* 37.9 (2005), pp. 1746–1751. DOI: https://doi.org/10.1016/j.biocel.2005.04.011.

[130] Zhu Xiaojin and Ghahramani Zoubin. 'Learning from labeled and unlabeled data with label propagation'. In: *Technical Report CMU-CALD-02–107*. Carnegie Mellon University, 2002.

[131] Zhenyue Zhang and Hongyuan Zha. 'Nonlinear Dimension Reduction via Local Tangent Space Alignment'. In: *Intelligent Data Engineering and Automated Learning*. Berlin, Heidelberg, 2003, pp. 477–481. DOI: https://doi.org/10.1007/978-3-540-45080-1_66.

Original manuscripts

Paper I A. Oslandsbotn, Z. Kereta, V. Naumova, Y. Freund and A. Cloninger, 'StreaMRAK a streaming multi-resolution adaptive kernel algorithm'. Published in *Applied Mathematics and Computation* (2022). (DOI: doi.org/10.1016/j.amc.2022.127112)

Paper II A. Oslandsbotn, A. Cloninger, and N. Forsch, 'Improving inversion of model parameters from action potential recordings with kernel methods'. Submitted to *Mathematical Biosciences and Engineering* (2023). (BioRxiv: doi.org/10.1101/2023.03.15.532862)

Paper III R. Bhattacharjee, A. Cloninger, Y. Freund, and A. Oslandsbotn 'Effective resistance in metric spaces'. Submitted to *Journal of Machine Learning Research* (2023). (Co-first-authors: Robi Bhattacharjee and Andreas Oslandsbotn) (arXiv: doi.org/10.48550/arXiv.2306.15649)

Paper IV R. Bhattacharjee, A. Cloninger, Y. Freund, and A. Oslandsbotn 'Structure from voltage'. In preparation for journal submission. (Co-first-authors: Robi Bhattacharjee and Andreas Oslandsbotn) (arXiv: doi.org/10.48550/arXiv.2203.00063)

Paper I

StreaMRAK a streaming multi-resolution adaptive kernel algorithm

Andreas Oslandsbotn, **Željko Kereta**
Valeria Naumova, **Yoav Freund**
Alexander Cloninger

Published in Applied Mathematics and Computation
(doi.org/10.1016/j.amc.2022.127112)

I

I

Applied Mathematics and Computation 426 (2022) 127112

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

StreaMRAK a streaming multi-resolution adaptive kernel algorithm☆,☆☆

Andreas Oslandsbotn[a,b,*], Željko Kereta[c], Valeriya Naumova[d], Yoav Freund[e], Alexander Cloninger[e]

[a] University of Oslo, Problemveien 7, Oslo 0315, Norway
[b] Simula School of Research and Innovation, Martin Linges Vei 25, Fornebu 1364, Norway
[c] University College London, Gower St, London WC1E 6BT, England
[d] Simula Research Laboratory, Martin Linges vei 25, Fornebu 1364, Norway
[e] University of California San Diego, 9500 Gilman Dr, La Jolla CA 92093, United States

ARTICLE INFO

Article history:
Received 15 August 2021
Revised 22 February 2022
Accepted 19 March 2022
Available online 9 April 2022

2021 MSC:
68Q32
65D15
46E22
68W27

Keywords:
Streaming
Reproducing kernel Hilbert space
Kernel methods
Laplcian pyramid
Adaptive kernel
Sub-sampling

ABSTRACT

Kernel ridge regression (KRR) is a popular scheme for non-linear non-parametric learning. However, existing implementations of KRR require that all the data is stored in the main memory, which severely limits the use of KRR in contexts where data size far exceeds the memory size. Such applications are increasingly common in data mining, bioinformatics, and control. A powerful paradigm for computing on data sets that are too large for memory is the *streaming model of computation*, where we process one data sample at a time, discarding each sample before moving on to the next one. In this paper, we propose StreaMRAK - a streaming version of KRR. StreaMRAK improves on existing KRR schemes by dividing the problem into several levels of resolution, which allows continual refinement to the predictions. The algorithm reduces the memory requirement by continuously and efficiently integrating new samples into the training model. With a novel sub-sampling scheme, StreaMRAK reduces memory and computational complexities by creating a *sketch* of the original data, where the sub-sampling density is adapted to the bandwidth of the kernel and the local dimensionality of the data. We present a showcase study on two synthetic problems and the prediction of the trajectory of a double pendulum. The results show that the proposed algorithm is fast and accurate.

1. Introduction

Machine learning algorithms based on kernel ridge regression (KRR) [1] is an active field of research [2–6], with applications ranging from time series prediction in finance [7], parameter inference in dynamical systems [8], to pairwise learning

☆ Code is found at: https://github.com/AndOslandsbotn/StreaMRAK.git
☆☆ Abbreviations: KRR (kernel regularized ridge regression), LP (Laplacian pyramid), CT (Cover-tree), DCT (Damped CT).
* Corresponding author.
E-mail addresses: andreas.oslandsbotn@gmail.com (A. Oslandsbotn), z.kereta@ucl.ac.uk (Ž. Kereta), valeriya@simula.no (V. Naumova), yfreund@eng.ucsd.edu (Y. Freund), acloninger@ucsd.edu (A. Cloninger).

https://doi.org/10.1016/j.amc.2022.127112

[9], face recognition [10] and drug estimation and gene analysis in biomedicine [11,12]. This paper develops a streaming variation of KRR using a radial kernel, a new sub-sampling scheme, and a multi-resolution formulation of the learning model.

Many popular data analysis software packages, such as Matlab^{TM} require loading the entire dataset into memory. While computer memory is growing fast, the size of available data sets is growing much faster, limiting the applicability of *in-memory* methods.[1]

Streaming [13] is a computational model where the input size is much larger than the memory size. Streaming algorithms read one item at a time, update their memory, and discard the item. The computer memory is used to store a *model* or a *sketch* of the overall data distribution, which is orders of magnitude smaller than the data itself. The development of streaming algorithms is experiencing increased popularity in the face of big data applications such as data mining [14] and bioinformatics [15], where data sets are typically too large to be kept *in-memory*. Many big data applications call for non-linear and involved models, and thus, the development of non-parametric and non-linear models is critical for successful learning.

Among the most popular non-parametric learning algorithms are kernel methods, which include well-known learning schemes such as the support vector machine (SVM) and KRR, to name a few. The appeal of kernel methods lies in their strong theoretical foundation [1,16], as well as their ability to map complex problems to a linear space without requiring an explicit mapping. A common class of kernels are radial kernels $k(\mathbf{x}, \tilde{\mathbf{x}}) = \Phi(\|\mathbf{x} - \tilde{\mathbf{x}}\|/r)$ for $\mathbf{x}, \tilde{\mathbf{x}} \in \mathcal{X} \subseteq \mathbb{R}^D$ and bandwidth $r > 0$ [17]. These kernels are universal (with a few exceptions [18]), meaning that they can approximate any bounded continuous function on $\mathcal{X}$ arbitrarily well. However, in high dimensions, kernel methods suffer from the "curse of dimensionality" and require large amounts of training data to converge. Furthermore, the computational complexity, memory requirement, and the number of parameters to learn grow unbounded with the number of training samples, a drawback known as the "curse of kernelization" [19]. In the context of streaming, the prospect of unbounded data streams makes this shortcoming even more detrimental.

Although kernel-based learning schemes are typically formulated as convex optimization problems, which do not require tuning hyper-parameters such as learning rate etc., there is still a need to determine the optimal kernel. For the Gaussian kernel, this amounts to selecting the bandwidth. Classically, an optimal kernel is chosen through batch techniques such as leave-one-out and k-fold cross-validation [20–22]. However, these approaches are inefficient as they spend significant time evaluating bad kernel hypotheses and often use multiple runs over the data, which is impossible in a streaming setting.

To meet a need for non-linear non-parametric algorithms for streaming data, we propose the *streaming multi-resolution adaptive kernel algorithm* (StreaMRAK) - a computationally and memory-efficient streaming variation of KRR. StreaMRAK address the kernel selection problem with a multi-resolution kernel selection strategy that adapts the sub-sample density to the kernel bandwidth over several levels of resolution. Furthermore, StreaMRAK addresses the curse of dimensionality and kernelization in a novel way, through the sub-sampling scheme.

1.1. Setting

We consider a finite sample data-cloud $\mathcal{X}$, $|\mathcal{X}| = n$, that is sampled i.i.d. according to a fixed but unknown distribution $\mathcal{P}$ over $\mathbb{R}^D$. The target is a bounded and continuous function $f : \mathbb{R}^D \to \mathbb{R}$. We assume that the points in $\mathcal{X}$ are placed in a *sequence* and that their order is random. [2] Each instance $\mathbf{x}_i \in \mathcal{X}$, for $i \in [n]$, paired with a label y_i where $y_i = f(\mathbf{x}_i) + \varepsilon_i$ and $\varepsilon_i \sim \mathcal{N}(0, \sigma)$ represents noise. The task of learning is to train a model $\widehat{f}$ that is a good approximation of the target function f.

In this work, we think about the intrinsic dimension of $\mathcal{X}$ as a local quantity, meaning it depends on the region $\mathcal{A} \subseteq \mathcal{X}$ and the radius r at which we consider the point cloud. To estimate the local intrinsic dimension in a "location and resolution sensitive" manner, we use the concept of the doubling dimension of a set, defined in Def. 1.2. See [23,24] for related definitions.

Definition 1.1 (Covering number)**.** Consider a set $\mathcal{A}$ and a ball $\mathcal{B}(\mathbf{x}, r)$, with $r > 0$ and $\mathbf{x} \in \mathcal{A}$. We say that a finite set $\mathcal{S} \subset \mathcal{B}(\mathbf{x}, r)$ is a covering of $\mathcal{B}(\mathbf{x}, r)$ in $\mathcal{A}$ if $\mathcal{A} \cap \mathcal{B}(\mathbf{x}, r) \subset \cup_{\mathbf{x}_i \in \mathcal{S}} \mathcal{B}(\mathbf{x}_i, r/2)$. We define the covering number $\kappa(\mathcal{A}, \mathbf{x}, r)$ as the minimum cardinality of any covering of $\mathcal{B}(\mathbf{x}, r)$ in $\mathcal{A}$.

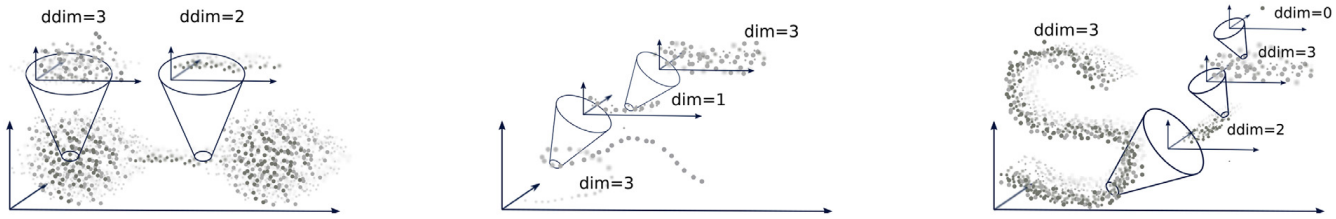
Definition 1.2 (Doubling dimension)**.** The doubling dimension $\texttt{ddim}(\mathcal{A}, r)$ of a set $\mathcal{A}$ is defined as $\texttt{ddim}(\mathcal{A}, r) = \lceil \log \kappa(\mathcal{A}, \mathbf{x}, r) \rceil$. For an interval $\mathcal{I} \subset \mathbb{R}_{>0}$ we define the doubling dimension as the least upper bound over $r \in \mathcal{I}$, that is $\texttt{ddim}(\mathcal{A}, \mathcal{I}) = \max_{r \in \mathcal{I}} \texttt{ddim}(\mathcal{A}, r)$.

We say that the intrinsic dimension of $\mathcal{X}$ changes with the location if there exist $\mathcal{A}_1, \mathcal{A}_2 \subset \mathcal{X}$ such that $\texttt{ddim}(\mathcal{A}_1, r) \neq \texttt{ddim}(\mathcal{A}_2, r)$ for $r > 0$. Similarly, we say that the intrinsic dimensionality of $\mathcal{X}$ changes with the resolution, if there exist $r_1 \neq r_2$ such that the doubling dimension $\texttt{ddim}(\mathcal{A}, r_1) \neq \texttt{ddim}(\mathcal{A}, r_2)$ for $\mathcal{A} \subset \mathcal{X}$.

In Fig. 1 we consider three examples to provide further insight on the doubling dimension. In Fig. 1a we see a domain shaped like a dumbbell, where the spheres are high dimensional, and the bar connecting them is lower-dimensional,

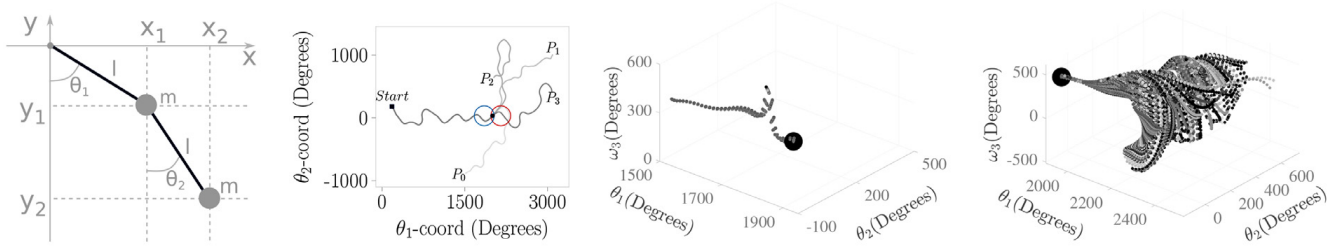
[1] Simulink^{TM}, a companion software to Matlab^{TM} supports streaming but has a much more limited computational model, targeted at signal processing applications.

[2] The assumption that the sequence is randomly ordered allows us to draw statistical conclusions from prefixes.



(a) Intrinsic dimension changes with location. (b) Noise has high intrinsic dimension. (c) Intrinsic dimension changes with the resolution.

Fig. 1. Three examples of variation in the intrinsic dimension. The coloring of the point clouds illustrates depth.



(a) Double pendulum (b) 2-dim projection of the pendulum training data $\mathcal{X}$. (c) 3-dim projection of the training data in the blue circle in (b). (d) 3-dim projection of the training data in the red circle in (b).

Fig. 2. (a) Illustration of a double pendulum. Here l and m are the length and mass of the pendulum rods, and θ_1, θ_2 are the angles. Furthermore, (x_1, y_1) and (x_2, y_2) are the positions of the point masses of the two pendulums. (b) Phase diagram of four double pendulums P_0, P_1, P_2, P_3, iterated for $T = 500$ time steps. The bifurcation point at step $T = 300$ is indicated with a black solid circle. (c) and (d) includes the ω_1 axis and zoom in on respectively the blue and red circles in (b). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

showing how the dimension can change with the location. Meanwhile, Fig. 1b illustrates a lower-dimensional manifold, embedded in $\mathbb{R}^3$, with manifold noise ζ_m. We see that when the resolution is sufficiently small, so that $r \approx \zeta_m$, the doubling dimensionality increases towards the dimension of the ambient space $\mathbb{R}^D$. Furthermore, Fig. 1c shows a point cloud that is locally 2-dimensional, but is embedded in a 3-dimensional space. By reducing r we can resolve this lower dimensionality, but if it is reduced further, we would eventually resolve the noise level, and the doubling dimension increases again.

As an example of how the intrinsic dimension might change with respect to regions and resolutions, we consider a double pendulum system, a well-known chaotic system that depends heavily on its initial conditions [25]. Systems with multiple pendulum elements are well known in engineering applications such as mechanical and robotic systems with several joints and are studied for their chaotic properties [26].

In Fig. 2b we visualize the trajectory of four pendulums P_0, P_1, P_2, P_3, for which the trajectories are indistinguishable until a bifurcation occurs around $T = 300$ time steps, and the trajectories start to diverge. In Fig. 2c and Fig. 2d we zoom in on the trajectory of all 500 pendulums in regions before and after the bifurcation. These two regions, $\mathcal{A}_1$ and $\mathcal{A}_2$, are indicated by a blue and red circle, respectively, in Fig. 2b. From the figures, it is clear that learning the trajectory in $\mathcal{A}_1$ is significantly easier than in $\mathcal{A}_2$, where learning the trajectory is more affected by the curse of dimensionality.

1.2. Contribution and comparison to related work

Contributions of this work can be divided into three components.

(C1) A multi-resolution variation of the state-of-the-art KRR solver FALKON [2], using the LP, which refines the predictions at each level of resolution by regressing on the errors from the previous level.
(C2) A novel sub-sampling scheme for kernel methods, tailored for use in combination with the LP, that can handle the curse of dimensionality and does not require the data to be *in-memory*.
(C3) Development of a streaming variation of FALKON, where the time and memory requirements depend on the doubling dimensionality and the level of resolution, instead of the number of training points. see Props. 5.3–5.5.

In the following, we give further details on these contributions and compare them to related work. The computational backbone of StreaMRAK is based on the state-of-the-art KRR solver FALKON [2], which among other things combines sub-sampling and preconditioning to process large data sets efficiently. However, FALKON relies on selecting an optimal kernel bandwidth, which can be inefficient in streaming.

Our first contribution (C1) addresses the issue of selecting an optimal bandwidth by introducing a multi-resolution reformulation of FALKON using a changing bandwidth variation of the LP [27–29]. This strategy is inspired by the success of existing multi-resolution approaches [4,30–36] and, for online learning, adaptive bandwidth approaches [37–39]. Our scheme combines the LP with a localized kernel, which gives a frequency and location-based discretization, similar to wavelet anal-

ysis that have shown great success in numerous applications. However, typical wavelet architectures [40–46] require upfront construction of a wavelet basis, which is not compatible with a data-adaptive kernel.

In this work, we aim to show that the LP is a viable multi-resolution scheme and can be modified to the streaming setting. Furthermore, we provide convergence bounds for the LP in the context of radial kernels and KRR, and show experimentally that it improves the estimation accuracy.

Let us now discuss our second contribution (C2). FALKON addresses the curse of kernelization by combining Nyström sub-sampling, conjugate gradient, and preconditioning, and achieves time and memory requirements of $\mathcal{O}(n\sqrt{n})$ and $\mathcal{O}(n)$ respectively, where n is the number of samples. In recent years there have been several efforts to address the curse of kernelization in similar ways through sub-sampling techniques such as sketching [3,5], randomized features [47–50] and Nyström sub-sampling [51–55]. However, despite their successes, these techniques are in principle *in-memory* type algorithms since they require access to the training data in advance of the training and are not optimized for streaming.

Furthermore, FALKON selects the sub-samples uniformly over the input domain $\mathcal{X}$. However, when learning with a radial kernel, the density of samples should be related to the bandwidth of the kernel. Otherwise, a too-small bandwidth will lead to bad interpolation properties, while a too-large bandwidth gives an ill-conditioned system [34]. Since the LP scheme reduces the kernel bandwidth at each level of resolution, it would be problematic to use the same sub-sample density. Furthermore, due to the curse of dimensionality, the covering number increases exponentially with the doubling dimension. Therefore, if doubling dimensionality varies across different regions of the domain $\mathcal{X}$, as illustrated by Fig. 1a, then the number of sub-samples necessary to maintain the density for a given bandwidth will also vary.

Our second contribution (C2) provides an alternative sub-sampling strategy, adapting the sub-sampling density to the kernel bandwidth. This strategy is based on a damped cover-tree (DCT), which is a modified version of the cover-tree (CT) [24], a tree-based data structure with $\mathcal{O}(n)$ memory and $\mathcal{O}(c^6 n \log n)$ time.

A problem with an adaptive sub-sampling strategy is its vulnerability to the curse of dimensionality. In regions of high doubling dimensions, the number of samples to achieve a certain density increases exponentially, as quantified by Def. 1.2. This means that the number of sub-samples from the CT will quickly grow too large for efficient computing. The danger is to waste resources on samples from subsets and levels where the doubling dimension is so large that good interpolation cannot be achieved for any viable sample sizes. This would only serve to slow down the computation and not increase the precision.

Due to this, the DCT introduces a damping property, which gradually suppresses the selection of sub-samples where the doubling dimensionality is large. This has the additional advantage of allowing to choose more sub-samples from regions where the doubling dimensionality is small. Thus, the DCT can diminish the impact of the curse of dimensionality. Furthermore, the DCT can be built continuously as new samples come in, making it ideal for a streaming computational model.

Our third contribution (C3), is the streaming capabilities of StreaMRAK. In particular, the sub-sampling and kernel construction allows for continuous integration of new training points. Furthermore, the DCT, the multi-resolution construction, and the KRR solver can all be multi-threaded and parallelized.

1.3. Organization of the paper

The paper is organized as follows. Section 2 introduces kernel methods and the FALKON algorithm, as well as the LP. Section 3 introduces the adaptive sub-sampling scheme and the DCT. StreaMRAK is described in Section 4 and an analysis of the algorithm is given in Section 5. Finally, Section 6 presents several numerical experiments and Section 7 gives an outlook for further work. The Appendix includes further mathematical background and the proofs.

1.4. Notation

We denote vectors $\mathbf{a} \in \mathbb{R}^D$ with boldface and matrices $\mathbf{A} \in \mathbb{R}^{n\times m}$ with bold uppercase, and $\mathbf{A}^\top$ denotes the matrix transpose. We use $\mathbf{K}_{nm}$ for kernel matrices, where the dimensionality is indicated by the subscripts. We reserve n for the number of training samples and m for the number of sub-samples. The ij-th element of a kernel matrix is denoted $[\mathbf{K}_{nm}]_{ij}$, while for other matrices we use $\mathbf{A}_{ij}$. The notation a_i indicates i-th element of a vector $\mathbf{a}$. Furthermore, we use $f([\mathbf{x}_n])$ to denote $(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n))^\top \in \mathbb{R}^n$, and $[m]$ to denote $\{i\}_{i=1}^m$. The notation $\mathbf{x}_i$ indicates the i-th training example. We use $\mathbf{a}^{(l)}$ and $\mathbf{A}^{(l)}$, where l refers to a specific level in the LP and the DCT. We take $\|\cdot\|$ to be the L^2 norm and $\|\cdot\|_{\mathcal{H}}$ to be the RKHS norm. We denote the intrinsic dimension of a manifold with d and the dimension of the embedding with D. By $\mathbb{1}_{\mathcal{S}}(\mathbf{x})$ we denote the indicator function, which evaluates to 1 if $\mathbf{x} \in \mathcal{S}$ and 0 otherwise, of a set $\mathcal{S} \subset \mathbb{R}^D$.

2. Kernel methods

Consider a positive definite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, defined on an input space $\mathcal{X} \subset \mathbb{R}^D$. Given data $\{(\mathbf{x}_i, y_i) : i \in [n]\}$ of samples from $\mathcal{X} \times \mathbb{R}^D$, kernel ridge regression computes an estimator by minimising

$$\widehat{f}_{n,\lambda} = \underset{f\in\widetilde{\mathcal{H}}}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n}(f(\mathbf{x}_i) - y_i)^2 + \lambda\|f\|_{\mathcal{H}}^2,$$

where $\mathcal{H}$ is the Hilbert space induced by the kernel. This allows to reduce the problem to a linear system

$$(\mathbf{K}_{nn} + \lambda n \mathbf{I}_n)\boldsymbol{\alpha} = \mathbf{y}, \text{ for } [\mathbf{K}_{nn}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j), \text{ and } \mathbf{y} = (y_1, \dots, y_n)^\top. \tag{2.1}$$

Coefficients $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n)^\top$ define the estimator by $f(\mathbf{x}) = \sum_{i=1}^n \alpha_i k(\mathbf{x}, \mathbf{x}_i)$. However, solving (2.1) using traditional methods has a time complexity of $\mathcal{O}(n^2)$, which can be costly for large n [2].

FALKON [2] addresses this issue by sub-sampling the columns of $\mathbf{K}_{nn}$, which reduces the effective complexity while maintaining accuracy. Namely, denote $\Gamma_n = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ and for $m \ll n$ let $\widetilde{\Gamma}_m = \{\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_m\}$ be Nyström centers (i.e. a randomly selected subset of Γ_n). Minimizing

$$\widehat{f}_{n,m,\lambda} = \underset{f \in \widetilde{\mathcal{H}}_M}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^n (f(\mathbf{x}_i) - y_i)^2 + \lambda \|f\|_{\mathcal{H}}^2, \tag{2.2}$$

where $\widetilde{\mathcal{H}}_m = \overline{\operatorname{span}}\{k(\cdot, \tilde{\mathbf{x}}_j) : j \in [m]\}$, leads to a linear system

$$\mathbf{H}\widetilde{\boldsymbol{\alpha}} = \mathbf{z}, \text{ for } \mathbf{H} = \mathbf{K}_{nm}^\top \mathbf{K}_{nm} + \lambda n \mathbf{K}_{mm}, \text{ and } \mathbf{z} = \mathbf{K}_{nm}\mathbf{y}.$$

Here $[\mathbf{K}_{nm}]_{ij} = k(\mathbf{x}_i, \tilde{\mathbf{x}}_j) \in \mathbb{R}^{n \times m}$ is the column-subsampled matrix and the estimator is given by $\widehat{f}_{n,m,\lambda}(\mathbf{x}) = \sum_{j=1}^m \widetilde{\alpha}_j k(\mathbf{x}, \tilde{\mathbf{x}}_j)$. To further reduce the time complexity FALKON uses a suitable preconditioner to reduce the condition number. The preconditioner is defined as $\mathbf{B}\mathbf{B}^\top = (n/m\mathbf{K}_{mm}^2 + \lambda n \mathbf{K}_{mm})^{-1}$, which is a natural (lower complexity) approximation of the ideal preconditioner $\mathbf{A}\mathbf{A}^\top = (\mathbf{K}_{nm}^\top \mathbf{K}_{nm} + \lambda n \mathbf{K}_{mm})^{-1}$. We now solve for $\widetilde{\boldsymbol{\alpha}}$ from the system of equations

$$\mathbf{B}^\top \mathbf{H} \mathbf{B} \boldsymbol{\beta} = \mathbf{B}^\top \mathbf{z}, \text{ for } \mathbf{H} = \mathbf{K}_{nm}^\top \mathbf{K}_{nm} + \lambda n \mathbf{K}_{mm}, \ \mathbf{z} = \mathbf{K}_{nm}\mathbf{y}, \text{ and } \widetilde{\boldsymbol{\alpha}} = \mathbf{B}\boldsymbol{\beta}. \tag{2.3}$$

This is solved iteratively, using the conjugate gradients with early stopping. Choosing $m = \mathcal{O}(\sqrt{n})$ still ensures optimal generalisation (i.e. same as KRR), while reducing the computational complexity to $\mathcal{O}(n\sqrt{n})$.

2.1. Streaming adaptation of FALKON

Matrices and vectors involved in the linear system in (2.3) can be separated into two classes: those that depend only on sub-samples in $\widetilde{\Gamma}_m$; and those ($\mathbf{K}_{nm}^\top \mathbf{K}_{nm}$ and $\mathbf{z}$) that also depend on all the training points Γ_n. Critically, terms in both groups are all of size m, which allows to reduce the complexity. Consider now the set of sub-samples $\widetilde{\Gamma}_m$ to be fixed, and assume new training points, in the form $\{(\mathbf{x}_q, y_q) : q = n+1, \dots, n+t)\}$, are coming in a stream. We can then update the second class of terms according to

$$\left[(\mathbf{K}_{(n+t)m})^\top \mathbf{K}_{(n+t)m}\right]_{ij} = \left[(\mathbf{K}_{nm})^\top \mathbf{K}_{nm}\right]_{ij} + \sum_{q=n+1}^{n+t} k(\mathbf{x}_q, \tilde{\mathbf{x}}_i) k(\mathbf{x}_q, \tilde{\mathbf{x}}_j), \tag{2.4}$$

$$\left[(\mathbf{K}_{(n+t)m})^\top \mathbf{y}\right]_i = z_i + \sum_{q=n+1}^{n+t} k(\mathbf{x}_q, \tilde{\mathbf{x}}_i) y_q. \tag{2.5}$$

Thus, only sub-samples $\widetilde{\Gamma}_m$, matrices $\left(\mathbf{K}_{nm}\right)^\top \mathbf{K}_{nm}$, $\mathbf{K}_{mm}$ and $\mathbf{z}$, need to be stored. However, in order to continuously incorporate new training points into Eqs. (2.4) and (2.5), sub-samples $\widetilde{\Gamma}_m$ must be determined in advance. Whereas this works if all the data is provided beforehand, it cannot be done if the data arrives sequentially. In this work, we address this through a multi-resolution framework. The overall estimator is composed of a sequence of estimators defined at different resolution levels of the domain. Correspondingly, the set of sub-samples $\widetilde{\Gamma}_m$ consists of smaller sets $\widetilde{\Gamma}^{(l)}_{m^{(l)}}$ that correspond to individual levels of resolution. The sets $\widetilde{\Gamma}^{(l)}_{m^{(l)}}$ are filled as the data streams in, and once a set for a given level is deemed complete, we proceed with updating (2.4) and (2.5).

Further details of how the sets $\widetilde{\Gamma}^{(l)}_{m^{(l)}}$ are constructed, and the corresponding criteria, are provided in Sections 3 and 4. We begin by describing the multi-resolution framework of estimators.

2.2. The laplacian pyramid

The LP [27,29] is a multi-resolution regression method for extending a model $\widehat{f}$ to out-of-sample data points $\mathbf{x} \in \mathcal{X}/\Gamma_n$. The LP can be formulated for radial kernels in the form

$$k(\mathbf{x}_i, \mathbf{x}_j) = \Phi\left(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{r}\right), \tag{2.6}$$

where $r > 0$ is a shape parameter that determines the decay of Φ with respect to $\|\mathbf{x}_i - \mathbf{x}_j\|$, see [17]. The idea underpinning the LP is to approximate the target function sequentially, where at each stage we regress on the errors from the previous stage. In other words, we begin with a rough approximation using a large shape parameter for which Φ decays slowly and

then improve the approximation by fitting the resulting error and reducing the shape parameter. In the LP, the estimator at level $L \in \mathbb{N}$ is defined recursively as

$$\widehat{f}^{(L)}(\mathbf{x}) = \sum_{l=0}^{L} s^{(l)}(\mathbf{x}) = s^{(L)}(\mathbf{x}) + \widehat{f}^{(L-1)}(\mathbf{x}), \tag{2.7}$$

where $\hat{f}^{(0)} = s^{(0)}$, and $s^{(l)}(\mathbf{x})$ is a correction term defined by

$$s^{(l)}(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i^{(l)} k^{(l)}(\mathbf{x}, \mathbf{x}_i). \tag{2.8}$$

The coefficients $\boldsymbol{\alpha}^{(l)} = (\alpha_1^{(l)}, \dots, \alpha_n^{(l)})^\top$ are computed by conducting KRR on the residuals, i.e. errors, from the estimator at the previous level. Namely, $\boldsymbol{\alpha}^{(l)} = (\mathbf{K}_{nn}^{(l)} + \lambda n \mathbf{I})^{-1} \mathbf{d}^{(l)}$, where

$$\mathbf{d}^{(l)} = \begin{cases} \mathbf{y}, & \text{if} \quad l = 0 \\ \mathbf{y} - \widehat{f}^{(l-1)}([\mathbf{x}_n]), & \text{otherwise} \end{cases}. \tag{2.9}$$

For a FALKON adaption of this scheme, we only need to modify how per-level coefficients are computed. Following (2.3) we iteratively solve

$$(\mathbf{B}^{(l)})^\top \mathbf{H}^{(l)} \mathbf{B}^{(l)} \boldsymbol{\beta}^{(l)} = (\mathbf{B}^{(l)})^\top \left(\mathbf{K}_{nm}^{(l)}\right)^\top \mathbf{d}^{(l)}, \tag{2.10}$$

where $\mathbf{B}^{(l)}$ is the corresponding preconditioner, and $\mathbf{H}^{(l)} = (\mathbf{K}_{nm}^{(l)})^\top \mathbf{K}_{nm}^{(l)} + \lambda n \mathbf{K}_{mm}^{(l)}$, and set $\widetilde{\boldsymbol{\alpha}}^{(l)} = \mathbf{B}^{(l)} \boldsymbol{\beta}^{(l)}$.

Remark 2.1. In this paper, we construct the kernel matrices $\mathbf{K}^{(l)}$ on a particular class of radial kernels, namely the Gaussian kernel

$$k^{(l)}(\mathbf{x}, \tilde{\mathbf{x}}_i) = \exp\left(- \frac{\|\mathbf{x} - \tilde{\mathbf{x}}_i\|^2}{2 r_l^2} \right),$$

where $r_l > 0$ is the shape parameter (the kernel bandwidth) at level l.

3. The damped cover tree

This work introduces a data-driven sub-sampling method that we call the damped cover-tree (DCT). The DCT is a modification of the cover-tree (CT) [24], a data structure based on partitioning a metric space, initially designed to facilitate nearest neighbor search. The goal of the DCT is to modify and simplify the CT to allow a viable sub-sampling scheme.

Let $(\mathcal{X}, \|\cdot\|)$ be a normed space where the input domain $\mathcal{X} \subset \mathbb{R}^D$ is bounded, such that the diameter $r_0 = \mathrm{diam}(\mathcal{X})$ is finite. The DCT is a tree structure where each node p of the tree is associated with a point $\mathbf{x}_p \in \mathcal{X}$, and which is built sequentially as data points arrive. Furthermore, let Q_l be a set (herein called a cover-set) containing all the nodes at a level $l \geq 0$ in the given tree. A level is associated with an integer l and a radius $r_l = 2^{-l} r_0$, where $l = 0$ denotes the root level containing only one node and l increases as we descend deeper into the tree. DCT has three invariants, of which the first two are also invariants of the CT.

(I1) (Covering invariant) For all $p \in Q_{l+1}$ there exists $q \in Q_l$ such that $\|\mathbf{x}_q - \mathbf{x}_p\| < r_l$.
(I2) (Separation invariant) For all $q, p \in Q_l$ where $\mathbf{x}_q \neq \mathbf{x}_p$, we have $\|\mathbf{x}_q - \mathbf{x}_p\| > r_l$.

We add that the standard CT includes a third invariant, the so-called nesting invariant, which requires $Q_l \subseteq Q_{l+1}$, but this is not desired for our purpose.

To introduce the last invariant of the DCT, we first need the following definition.

Definition 3.1 (The covering fraction)**.** Let $p \in Q_l$ be a node, and $\mathbf{x}_p$ the associated point in $\mathcal{X}$. Furthermore, let $\widetilde{C}_p = \{c_i\}_{i=1}^k$ be the children of p, and $\mathbf{x}_{c_i}$ the corresponding points in $\mathcal{X}$. The covering fraction of a node p is defined as

$$\mathfrak{cf}(p) = \frac{\mathrm{Vol}\left(\mathcal{X} \cap \mathcal{B}(\mathbf{x}_p, r_l) \cap \bigcup_{c_i \in \widetilde{C}_p} \mathcal{B}(\mathbf{x}_{c_i}, r_{l+1}) \right)}{\mathrm{Vol}\left(\mathcal{X} \cap \mathcal{B}(\mathbf{x}_p, r_l) \right)}.$$

The covering fraction is the proportion of the volume of $\mathcal{B}(\mathbf{x}_p, r_l)$ that is covered by balls around its children of half the radius. This quantity is directly related to (I2), which enforces the radius r_l to reduce by a factor of 2 for each new level, starting from an initial radius $r_0 > 0$. The covering fraction allows us to capture the vulnerability of the standard CT to the curse of dimensionality.

For example, consider two regions $\mathcal{A}_1, \mathcal{A}_2 \subseteq \mathcal{X}$, for which the doubling dimension at radius r_l is $\mathtt{ddim}(\mathcal{A}_1, r_l) > \mathtt{ddim}(\mathcal{A}_2, r_l)$. A node $p \in \mathcal{A}_1$ at level l will then need exponentially more children to be covered, than a node $q \in \mathcal{A}_2$ at the same level l. This exacerbates the deeper we go into the tree. Therefore, the CT would have significantly more nodes from regions where the doubling dimension is large.

We recall now that sub-sampling is in kernel methods intended to reduce the computational complexity. For this purpose, it is desirable to keep the number of sub-samples from each level within a budget of reasonable size. On the other hand, a too low sub-sample density will lead to poor interpolation performance. Due to the exponential growth of the number of nodes with respect to the doubling dimension, it would be desirable to avoid wasting our budget on sub-samples from regions and radii with a large doubling dimension, as this would require dedicating an (exponentially) large number of points to achieve good interpolation, which is not feasible. Moreover, in high dimensional regions, we likely cannot learn anything more than a simple function, for which a lower sampling density would suffice.

To reduce the number of sub-samples from regions of large doubling dimensionality, we introduce the following damping invariant as the third invariant of the DCT.

(I3) (Damping invariant) Let $\mathcal{D}_{\mathfrak{cf}} \in (0, 1)$ be some threshold and let $\widetilde{C}_p$ and $\mathfrak{cf}(p)$ be as in Def. 3.1. Then any node q whose parent node p does not satisfy $\mathfrak{cf}(p) \geq \mathcal{D}_{\mathfrak{cf}}$ does not have children of its own.

The damping invariant forces the tree to devote more resources to regions of lower doubling dimension by making it harder for nodes in regions with higher doubling dimensions to have children. In other words, the practical effect of the damping invariant is to stop the vertical growth of the DCT if the doubling dimension becomes large. This is because the covering number grows exponentially with the dimensionality, ensuring $\mathfrak{cf}(p) \geq \mathcal{D}_{\mathfrak{cf}}$ gets correspondingly harder to achieve.

Remark 3.2. In Section 5.1, we analyze the damping invariant in more detail and show how the damping suppresses vertical growth of the DCT more for regions of high doubling dimension than for regions of lower doubling dimensionality.

3.1. Construction of the DCT

We now discuss how the DCT is constructed and updated as the data streams in. First, it is important to restate that we use the DCT to replace the Nyström sampling, which was in FALKON used to reduce the complexity of the ridge regressor. Consequently, not all of the streamed data (that is, not every training point) will be added to the tree, but only those whose inclusion into the tree would not violate the invariants (I1)-(I3). In other words, the tree consists of only those training points that help resolve the data space at the relevant resolution level. Thus, each node p in the DCT is associated with a unique training sample $\mathbf{x}_p$, but not every training sample will be represented by a node in the tree. Note that this is different from the standard CT, which aims to organize all of the training data into a geometrical leveled data structure.

The construction of the DCT consists of a series of checks which examine whether adding a given data point to the DCT would, or would not, violate invariants (I1)- (I3). When a new point $\mathbf{x}_q$ arrives from the data stream the goal is to identify the deepest level l for which there exists a node p such that $\|\mathbf{x}_q - \mathbf{x}_p\| \leq r_l$. This corresponds to finding the nearest node in the tree that could serve as a parent.

We achieve this in the following way. The first training point is identified as the root node to which we associate the radius r_0. For each new point, we proceed in a top-down manner, starting from the root node[3]. We then check whether $\mathbf{x}_q$ would violate the separation invariant at the next level. In other words, if there exists a node p such that $\|\mathbf{x}_q - \mathbf{x}_p\| < r_l$. If such a node does not exist, then $\mathbf{x}_q$ is added to the set of children of the root node, and we update the covering fraction estimate for the root node. Otherwise, if such a node does exist, we repeat the process, checking the separation invariant among the children of the corresponding node, and proceed further down the tree.

Assume we arrived to a node p at level $l \geq 1$, and we have $\|\mathbf{x}_q - \mathbf{x}_p\| \leq r_l$. We then check if p is allowed to have children, that is if the damping invariant is satisfied. If it is not satisfied, the point $\mathbf{x}_q$ is dismissed (it is not added to the tree). On the other hand, if p is allowed to have children, we check whether the separation invariant holds, i.e., if there exists a child c of the node p such that $\|\mathbf{x}_q - \mathbf{x}_c\| < r_{l+1}$. If that were the case, the separation invariant would be violated, and the recursion is applied again by considering c as the potential parent node. However, if such a child does not exist, that is, if the separation invariant is not violated, then $\mathbf{x}_q$ is added to the set of children of the node p. More details are given in Alg. A.1.

Some comments are needed to elucidate how are the steps described above applied in practice. First, note that the covering fraction from Def. 3.1 cannot be calculated explicitly, since the volume terms require knowing the intrinsic dimensionality. Therefore, it is necessary to use an estimator instead. For this purpose, we interpret $\mathfrak{cf}(p)$ as the probability that a sample $\mathbf{x} \sim \mathrm{Uni}(\mathcal{B}(\mathbf{x}_p, r))$ will be within $\mathcal{B}_c := \bigcup_{c_i \in \widetilde{C}_p} \mathcal{B}(\mathbf{x}_{c_i}, r/2)$, where $\widetilde{C}_p$ are the children of p. This probability can be estimated by considering the checks of the separation invariant (I2), conducted on the last N points that were inside $\mathcal{B}(\mathbf{x}_p, r)$, as a series of independent random trials. We use the following running average as an estimator of the covering fraction

$$(\mathfrak{cf}(p))_t = (1-\alpha)(\mathfrak{cf}(p))_{t-1} + \alpha \mathbb{1}_{\mathcal{B}_c}(\mathbf{x}_t), \tag{3.1}$$

where $\mathbb{1}_{\mathcal{B}_c}(\mathbf{x}_t)$ is the indicator function, and $\alpha > 0$ is a weighting parameter. This approximates a weighted average of the outcome of the N last draws (cf. Appendix B). Note that this reduces the memory requirements, since instead of storing N trial outcomes for each node in the tree, as required had we used an average of the last N trials, we store only a single value for each node in the tree.

Second, the separation invariant is in practice too strict since it results in too few points added to the tree, and thus a worse kernel estimator. Moreover, checking the separation invariant adds to the computational complexity. Therefore, we

[3] We assume that all new points $\mathbf{x}_q$ are within a ball of radius r_0 around this node, which holds for a large enough r_0

introduce the following relaxation. Assume we have a new point $\mathbf{x}_q$ and arrived at a node p at level l. We then first conduct a random Bernoulli trial, with the failure probability

$$q_{\mathbf{x}} = \frac{1}{1+\exp\left[h\tan\left(\pi\left(\|\mathbf{x}_q-\mathbf{x}_p\|/r_l-\frac{1}{2}\right)\right)\right]}, \tag{3.2}$$

where h is the hardness of the threshold. In other words, the probability of failure is proportional to the distance between $\mathbf{x}_q$ and $\mathbf{x}_p$ - the larger the distance, the more likely the failure. If the trial's outcome is a failure, then the check for the separation invariant is ignored, and the algorithm continues. If it is a success, we proceed by first checking the separation invariant. This means that the probability to ignore the separation invariant increases as $\mathbf{x}_q$ gets farther from $\mathbf{x}_p$.

3.2. Sub-sampling from the DCT

We now discuss how the DCT is used for sub-sampling the training points. By organizing the training points into cover-sets Q_l the DCT allows a hierarchical sub-sampling. Even though cover-sets Q_l significantly reduce the number of training points, they are for practical purposes still too large for efficient sub-sampling. Due to this, we restrict ourselves to a subset $\widetilde{\Gamma}^{(l)} \subseteq Q_l$ of candidate sub-samples called landmarks.

Definition 3.3 (Landmarks)**.** Let Q_l be the cover-set at level l in a DCT. We define the set of candidate landmarks at level l as $\widetilde{\Gamma}^{(l)} = \{\mathbf{x}_p \mid p \in Q_l \text{ and } \mathfrak{cf}(p) \geq \mathcal{D}_{\mathfrak{cf}}\}$, and the set of landmarks (of size m) as any subset $\widetilde{\Gamma}^{(l)}_m = \{\tilde{\mathbf{x}}^{(l)}_1, \ldots, \tilde{\mathbf{x}}^{(l)}_m\} \subset \widetilde{\Gamma}^{(l)}$ of size m.

Some remarks are in order. First, by Def. 3.3, candidates for landmarks at level l are only those nodes allowed to have children (according to the damping invariant (I3)). This design choice implies that the set of candidate landmarks will contain more points from regions with a lower doubling dimension than points from regions with a higher doubling dimension. This is because the larger the doubling dimension is, the more children nodes are needed to cover a given parent node.

Second, Def. 3.3 suggests using only a subset of candidate landmarks as sub-samples. We refer to a result from [2] which states that good statistical accuracy of the estimator is achieved if the number of sub-samples is proportional to the square root of the number of samples. At level l we therefore use a set of landmarks which is of size $m^{(l)} = \delta_0\sqrt{|Q_l|}$, where $\delta_0 > 0$ is a constant.

The third point that requires attention concerns the question of when the landmarks should be selected. To that end, we use the covering fraction of a level, which, with a slight abuse of notation, we denote as $\mathfrak{cf}(Q_l)$. Moreover, we compute $\mathfrak{cf}(Q_l)$ as

$$(\mathfrak{cf}(Q_l))_t = (1-\alpha)(\mathfrak{cf}(Q_l))_{t-1} + \alpha\mathbb{1}_{\mathcal{B}_{\text{level}}}(\mathbf{x}_t), \tag{3.3}$$

where $\mathcal{B}_{level} = \bigcup_{p\in Q_l} \mathcal{B}(\mathbf{x}^{(l)}_p, r_l)$. Moreover, analogously to the damping invariant, let $\mathcal{D}_{level} \in (0,1)$ be some threshold. We then say that a level l is sufficiently covered when $\mathfrak{cf}(Q_l) \geq \mathcal{D}_{level}$.

Remark 3.4. We note that as the level increases, our estimate of $\mathfrak{cf}(Q_l)$ through Eq. (3.3) will be increasingly more sensitive to subsets $\mathcal{A} \subset \mathcal{X}$ of low doubling dimension than to subsets of large doubling dimension. This is because the damping invariant (I3) makes it harder for nodes in high dimensions to have children. Consequently, we will have fewer points in deeper levels that belong to high dimensional regions. Because of this, the estimator in Eq. (3.3) is biased towards using more sub-samples from lower dimensional regions.

Sub-sampling from a level l goes as follows. As training points arrive, we build the tree and continuously update the covering fraction of a level. Once that level is sufficiently covered, that is, once $\mathfrak{cf}(Q_l) \geq \mathcal{D}_{level}$, we extract the set of landmarks by sub-sampling $m^{(l)}$ points from the pool of candidate landmarks $\widetilde{\Gamma}^{(l)}$.

4. StreaMRAK

In this section, we present StreaMRAK and clarify how it synthesizes concepts from Sections 2 and 3, and utilizes them in a streaming context. The workflow of StreaMRAK can be divided into three threads that can run in parallel, subject to some inter-dependencies. These are the sub-sampling thread, the training thread, and the prediction thread. Overviews of these threads are given next, and the reader is referred to Algorithm A.2 in the Appendix for further details.

4.1. Sub-sampling thread

In the sub-sampling thread StreaMRAK collects and organizes the training data into a DCT. Namely, as new training pairs are collected, the covering (I1) and separation (I2) are checked, and the covering fraction is updated as described in Section 3.1. Moreover, the set of landmarks for each level is updated, as described in Section 3.2. Once the set of landmarks for a given level $\widetilde{\Gamma}^{(l)}_m$ is completed, the landmarks and the estimator for the corresponding level can be used in the remaining two threads.

4.2. Training thread

The model is trained at level l when two conditions are met. First, coefficients of the previous level $l-1$ in the LP must have been calculated, i.e. previous training thread must finish. Second, landmarks $\widetilde{\Gamma}^{(l)}_{m^{(l)}}$ at level l must be ready.

In the first step, we define the kernel matrix on the landmarks by

$$[\mathbf{K}^{(l)}_{mm}]_{ij} = k^{(l)}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j), \text{ for } \tilde{\mathbf{x}}_i \in \widetilde{\Gamma}^{(l)}. \tag{4.1}$$

In the second step we consider $\left(\mathbf{K}^{(l)}_{nm}\right)^\top \mathbf{K}^{(l)}_{nm} \in \mathbb{R}^{m^{(l)}\times m^{(l)}}$ and $\left(\mathbf{K}^{(l)}_{nm}\right)^\top \mathbf{d}^{(l)}_n \in \mathbb{R}^{m^{(l)}}$ which in addition to landmarks depend on the training points. They are updated continuously as new training points come in, according to Eq. (2.4) and Eq. (2.5). However, they are not updated indefinitely, but only until new training points do not significantly alter the matrices according to the following criterion.

Definition 4.1. (Sufficient training points) Let $\mathbf{A}_n := (\mathbf{K}^{(l)}_{nm})^\top \mathbf{K}^{(l)}_{nm}$, and $\mathbf{b}_n := \left(\mathbf{K}^{(l)}_{nm}\right)^\top \mathbf{d}^{(l)}_n$. Let $\delta_1, \delta_2, \delta_3 > 0$ be three constants. We consider the number of training points at a level l sufficient when either $n \geq \delta_3$ or

$$\left\| \frac{\mathbf{A}_n}{n} - \frac{\mathbf{A}_{n+1}}{n+1} \right\|_\infty \leq \delta_1 \text{ and } \quad \left\| \frac{\mathbf{b}_n}{n} - \frac{\mathbf{b}_{n+1}}{n+1} \right\| \leq \delta_2.$$

After enough training samples are collected according to Def. 4.1, the correction term $s^{(l)}$ is obtained by solving for the coefficients $\widetilde{\alpha}^{(l)}_1, \ldots, \widetilde{\alpha}^{(l)}_{m^{(l)}}$ using Eq. (2.10). The new prediction model $\widehat{f}^{(L)}$ is obtained by adding $s^{(l)}$ to the previous model, according to Eq. (2.7).

4.3. Prediction thread

In this thread StreaMRAK makes provides the latest version of the trained LP model in Eq. (2.7). This means that if L is currently the highest level that has been trained, the prediction for new points $\mathbf{x}$ is made using the model $\widehat{f}^{(L)}(\mathbf{x})$.

5. Analysis

In this section, we first analyze the damping invariant of the DCT. We then offer theoretical results on the convergence properties of the LP in the context of KRR. Finally, we offer estimates of the time and memory requirements of StreaMRAK. We introduce two auxiliary results.

Lemma 5.1. *Consider a ball $\mathcal{B}(\mathbf{x}, r) \in \mathbb{R}^D$ and let $\delta > 0$. The number of points in any (discrete) set of points within $\mathcal{B}(\mathbf{x}, r)$ that are at least δ apart, $S = \{\mathbf{x}_i \in \mathcal{B}(\mathbf{x}, r) | d(\mathbf{x}_i, \mathbf{x}_j) \geq \delta \text{ for } i \neq j\}$, is bounded by $|S| \leq \left(\frac{2r}{\delta} + 1\right)^D$.*

Proof. Since the points in S are at least δ apart, it follows that the balls $\mathcal{B}(\mathbf{x}_i, \delta/2)$ are disjoint. Consider now the ball $\mathcal{B}(\mathbf{x}, r + \delta/2)$. All of the balls $\mathcal{B}(\mathbf{x}_i, \delta/2)$ are entirely contained within $\mathcal{B}(\mathbf{x}, r + \delta/2)$. Since the balls $\mathcal{B}(\mathbf{x}_i, \delta/2)$ are disjoint, it follows that

$$|S| \leq \text{Vol}\left(\mathcal{B}(\mathbf{x}, r + \delta/2)\right) / \text{Vol}\left(\mathcal{B}(\mathbf{x}_i, \delta/2)\right) = \left(\frac{2r}{\delta} + 1\right)^D.$$

□

Lemma 5.2. *Consider a domain $\mathcal{X} \in \mathbb{R}^D$, a ball $\mathcal{B}(\mathbf{x}_p, r) \subset \mathcal{X}$ and let $S = \{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{B}(\mathbf{x}_p, r) | \|\mathbf{x}_i - \mathbf{x}_j\| \geq \delta \text{ for } i \neq j\}$. Furthermore, let the doubling dimension of the set S be $\texttt{ddim} := \texttt{ddim}(S, r)$. We let $c_d := |S|$ when $\texttt{cf}(p) = 1$. We then have $2^{\texttt{ddim}-1} \leq c_d \leq 5^{\texttt{ddim}}$.*

Proof. The upper bound on c_d follows from Lemma 5.1 with $r = r_0$ and $\delta = r_0/2$. The lower bound follows from the definition of the doubling dimension 1.2. □

5.1. Analysis of the DCT

As discussed in Section 3, the DCT adds a given training point to the set of nodes of the tree if conditions (I2) and (I3) are satisfied, and the points are otherwise discarded. In particular, the damping invariant (I3) makes it harder for a node to have children. The guiding idea is that damping should reduce the impact of the curse of dimensionality by making it harder for nodes in regions of higher doubling dimension to have children, and in doing so it should effectively stop the vertical growth of the tree in corresponding regions. Therefore, it is critical to understand how and to what degree the damping affects high dimensional regions more than low dimensional ones.

In a statistical sense, the damping should treat all nodes in regions of the same doubling dimension equally. Therefore, to gain insight into the damping, it suffices to analyze its effects concerning the doubling dimension on a single node p. In this case, the effect of damping can be measured by analyzing how many training points must pass through p, in the sense of Alg. A.1, before children of p are allowed to have children of their own. This can be modeled by considering the expected number of training points $\mathbf{x}_i \sim \text{Uni}\left(\mathcal{B}(\mathbf{x}_p, r)\right)$ necessary to cover $\mathcal{B}(\mathbf{x}_p, r)$ with balls of radius $r/2$ around points $\mathbf{x}_i$.

Consider $\mathbf{x}_i \sim \text{Uni}\big(\mathcal{B}(\mathbf{x}_p, r)\big)$, and let a set $\mathcal{S}_p$ be built in a succession of trials $i = 1, \dots, N_t$ so that

$$\mathbf{x}_i \in \mathcal{S}_p \text{ if } \|\mathbf{x}_i - \mathbf{x}\| \geq \frac{r}{2} \text{ for all } \mathbf{x} \in \mathcal{S}_p.$$

In other words, a newly sampled point $\mathbf{x}_i$ will only be added to the set $\mathcal{S}_p$ if it its pairwise distances from all the points that are already in $\mathcal{S}_p$ are at least $r/2$.

Problem 1. Let $\widetilde{C}_p$ denote the set of children of the node p, constructed from the above-described trials. What is the expected number of trials N_t needed to ensure $\mathfrak{c}\mathfrak{f}(p) = 1$?

Since there is no unique set $\mathcal{S}_p$ such that the corresponding set of children $\widetilde{C}_p$ ensures $\mathfrak{c}\mathfrak{f}(p) = 1$, the sample space for Problem 1 corresponds to all admissible sets $\mathcal{S}_p$, which vary in both the number and the location of points they contain. Characterizing all such sets corresponds to a disordered sphere packing problem [56], which is an NP-hard combinatorial problem [57]. For a theoretical analysis of this problem, defining a probability measure over the sample space is necessary. However, in this level of generality, neither the sample space nor the probability measure admit a workable definition, with currently available mathematical tools [56]. Although some theoretical insights are possible under simplifications on the sample space, this analysis is restrained to a limited number of spheres and configurations.

Due to these difficulties, we consider a simplified setting where we instead consider an average case. If the set $\mathcal{S}_p$ is such that $\mathcal{B}(\mathbf{x}_i, r) \subset \bigcup_{\mathbf{x}_i \in \mathcal{S}_p} \mathcal{B}(\mathbf{x}_i, r/2)$, which corresponds to $\mathfrak{c}\mathfrak{f}(p) = 1$, then each of the balls $\mathcal{B}(\mathbf{x}_i, r/2)$ occupies on average $\frac{1}{|\mathcal{S}_p|}$ of the total volume of $\mathcal{B}(\mathbf{x}_p, r)$, assuming none of the balls are covered by a union of other balls. Therefore, as $\mathcal{S}_p$ is being built, adding a point to $\mathcal{S}_p$ will, on average, reduce the unoccupied volume of $\mathcal{B}(\mathbf{x}_p, r)$ by $\frac{1}{|\mathcal{S}_p|}$. Moreover, it can be shown that the number of elements in such a set satisfies $2^{\texttt{ddim}-1} \leq |\mathcal{S}_p| \leq 5^{\texttt{ddim}}$, see Lemma 5.2, where $\texttt{ddim} := \texttt{ddim}(\mathcal{S}_p, r)$ is the doubling dimension of $\mathcal{S}_p$. Based on these considerations we introduce a simplified setting for the average case of Problem 1.

Assumption 1. Problem 1 can be approximated by dividing the ball $\mathcal{B}(\mathbf{x}_p, r)$ into a union of c_d fixed (and known) disjoint bins $\mathcal{B}_i$ of size $(1/c_d)\,\text{Vol}\big(\mathcal{B}(\mathbf{x}_p, r)\big)$.

Note that the bins referred to in Assumption 1 correspond to regions around the children of the node p. Assumption 1 reduces the average case of Problem 1 to a form of the classical coupons collector's problem [58], which considers n coupons with the same probability of being drawn. Through a series of randomized trials with replacement, the goal is to obtain a copy of each coupon. Relevant for Problem 1 is estimating the stopping time T, which counts the number of trials before all coupons are collected, and which satisfies $\mathbb{E}[T] = nH_n$, where n denotes the number of coupons and H_n is the n-th harmonic number [58].

In terms of Problem 1, and under Assumption 1, we can therefore identify $T = N_t$, $n = |\mathcal{S}_p|$ and $\mathbb{E}[N_t|\text{Node } p] = |\mathcal{S}_p| H_{|\mathcal{S}_p|}$. Combining the bound $\ln(n) + \frac{1}{2} \leq H_n \leq \ln(n) + 1$ (from [59]), with the bound on $|\mathcal{S}_p|$ from Lemma 5.2 we have

$$2^{\texttt{ddim}-1}((\texttt{ddim} - 1)\ln 2 + 1/2) \leq \mathbb{E}[N_t|\text{Node } p] \leq 5^{\texttt{ddim}}(\texttt{ddim}\ln 5 + 1). \tag{5.1}$$

With the same strategy, we can bound the number of trials until the cover-fraction of a level reaches 1, as

$$2^{l(\texttt{ddim}-1)}(l(\texttt{ddim} - 1)\ln 2 + 1/2) \leq \mathbb{E}[N_t|\text{Level } l] \leq 5^{l\texttt{ddim}}(l\texttt{ddim}\ln 5 + 1). \tag{5.2}$$

From Eq. (5.1) we see that the number of training points $\mathbb{E}[N_t|\text{node } p]$ grows exponentially with the doubling dimensionallity d. In other words, significantly more trials are needed to achieve $\mathfrak{c}\mathfrak{f}(p) = \mathcal{D}_{\mathfrak{c}\mathfrak{f}}$ for nodes in regions with a large doubling dimension than it is for nodes in regions with a lower doubling dimension. Consequently, through the damping invariant, the DCT restricts the vertical growth of the tree comparatively more the higher the doubling dimension of the local region.

5.2. Time and memory requirements

This section analyzes the memory requirements of StreaMRAK, which involve storing the DCT and the linear system components used to update the coefficients. Furthermore, we consider the computational requirements, which consist in solving the coefficient equations. Both the memory and computational requirements need to be analyzed per level l of the tree due to the multi-resolution nature of the estimator and the tree organization of the data.

For the analysis, we consider a simplified setting where we assume that the doubling dimension is constant for all levels and all subsets of $\mathcal{X}$, and that the number of children c_d is the same for all nodes. At the end of the section we describe a more general setting.

In the following, we assume that the growth of the DCT stops at a level L. In other words, level L is the last level at which there are nodes. In practice, the growth of the DCT slows down exponentially fast with the product of the doubling dimension $\texttt{ddim} := \texttt{ddim}(\mathcal{X}, r_L)$ and the level l. This can be seen from Eq. (5.2), which shows that the number of training points necessary to fill up a level grows exponentially with $l\texttt{ddim}$. Therefore, in practice, no new levels will be added to the DCT when $l\texttt{ddim}$ is large enough, which effectively makes the last level L independent of the number of training points.

Furthermore, from Lemma 5.2 we know that c_d is bounded by $2^{\texttt{ddim}-1} \leq c_d \leq 5^{\texttt{ddim}}$, which shows that also c_d is independent of the number of training points.

Proposition 5.3. *The memory requirement of StreaMRAK is* $\mathcal{O}\big(\sum_{l=0}^{L} c_d^l\big)$.

Proof. The memory requirement of the DCT is determined by the number of nodes in the tree. Given that the number of children is the same for all nodes. If the number of children per node is c_d, then the total number of nodes at level l is c_d^l. Thus, the memory needed to store the DCT with L levels is $\mathcal{O}(\sum_{l=0}^{L} c_d^l)$.

To store the linear system on level l we need the matrices $\big(\mathbf{K}_{nm^{(l)}}\big)^\top \mathbf{K}_{nm^{(l)}}$, $\mathbf{K}_{m^{(l)}m^{(l)}} \in \mathbb{R}^{m^{(l)} \times m^{(l)}}$ and the vector $\mathbf{z} \in \mathbb{R}^{m^{(l)}}$. The number of landmarks $m^{(l)}$ at level l is chosen as $m^{(l)} = \delta_0 \sqrt{|Q_l|}$, where $|Q_l|$ is the number of nodes at level l. Since $|Q_l|$ is $\mathcal{O}(c_d^l)$, it follows that $m^{(l)} \times m^{(l)}$ is also $\mathcal{O}(c_d^l)$ per level, and the desired conclusion follows. □

Note that with a fixed L and n larger than $\mathcal{O}\big(\sum_{l=0}^{L} c_d^l\big)$, then the memory requirement is independent of n. We also note that if the deepest level satisfies $L \to \infty$, then the number of nodes is determined by the number of training points, and the memory requirement would thus, in the worst case, become $\mathcal{O}(n)$, the same as for the standard cover-tree.

Next, we discuss the construction of the DCT, where adding a new point to the set of nodes requires a search through the tree.

Proposition 5.4. *Inserting a new point into the DCT, cf. Algorithm A.1, requires* $\mathcal{O}(c_d L)$ *operations.*

Proof. For a point $\mathbf{x}_q \in \mathcal{X}$ to be analyzed at level L, we need to have analyzed it at the previous $l < L$ levels. At each level, we must, in the worst case, check the separation invariant with all children of the current potential parent $p^{(l)}$, before finding a node c such that $\|\mathbf{x}_q - \mathbf{x}_c\| \leq 2^{-l} r_0$, that would serve as the next potential parent. This requires at most c_d operations per level, giving Lc_d total operations over the L levels. The same number of operations is necessary if a node is discarded at level L. □

Lastly, we analyze the computational requirements for solving the linear system.

Proposition 5.5. *The time requirement for solving the linear system in Eq. (2.3) is* $\mathcal{O}\big(\delta_3 m^{(l)} + \big(m^{(l)}\big)^3\big)$ *per level, where* δ_3 *is given in Def. 4.1,*

Proof. The time requirement of FALKON is $\mathcal{O}(nmt + m^3)$ where n is the number of training points, m the number of landmarks and t the number of iterations of the conjugate gradient (which has an upper bound). By Def. 4.1, StreaMRAK uses at most δ_3 training samples at each level. Since $m^{(l)}$ is the number of landmarks at level l, the result follows. □

Assume that the domain $\mathcal{X}$ can be divided into disjoint subsets $\mathcal{A}_1, \ldots, \mathcal{A}_t \subset \mathcal{X}$ for which the doubling dimension $\texttt{ddim}(\mathcal{A}_i, r_l)$ differs based on $\mathcal{A}_i$ and radius r_l. Let the number of children of a node $\mathbf{x}_p \in \mathcal{A}_i$ at level l be $c_{d,i,l}$. In this scenario, the growth of the DCT will stop at different levels L_i for different subsets $\mathcal{A}_i$. The final time and memory requirements would therefore be the sum of the contribution from each subset $\mathcal{A}_i$. In other words, the memory would be $\mathcal{O}(\sum_{i=1}^{t} \sum_{l=0}^{L_i} c_{d,i,l}^l)$, and similarly the time requirement per point insertion would be $\mathcal{O}(\sum_{i=1}^{t} \sum_{l=0}^{L_i} c_{d,i,l})$. We note that $c_{d,i,l}$ and L_i depend on the dimensionality of the data, but are independent of n. Therefore, so are the time and memory requirements.

5.3. Convergence of the LP formulation of the KRR

This section analyzes the conditions for which the LP approximates the training data $y_i = f(\mathbf{x}_i)$, with respect to the number of levels. A similar analysis was previously done for the LP in the context of kernel smoothers [28]. However, to the best of our knowledge, this is the first time the LP formulation of KRR has been analyzed in this way.

Theorem 5.6. *Let* $\widehat{f}^{(l)}$ *be the LP estimator defined in Eq. (2.7) and let* λ *be a regularization parameter. Furthermore, let* $0 < \sigma_{l,n} \leq \cdots \leq \sigma_{l,1}$ *be the eigenvalues of* $\mathbf{K}_{nn}^{(l)}$. *For* $L > 0$ *we then have*

$$\|\widehat{f}^{(L+1)}([\mathbf{x}_n]) - f([\mathbf{x}_n])\| \leq \prod_{l=0}^{L} (1 - \varepsilon(l)) \|\widehat{f}^{(0)}([\mathbf{x}_n]) - f([\mathbf{x}_n])\|, \quad \text{where} \quad \varepsilon(l) = \frac{\sigma_{l,n}}{n\lambda + \sigma_{l,n}}.$$

Proof. From the recurrence relationship for the residuals $\mathbf{d}^{(l)}$ in Eq. (2.9) it follows by induction that

$$\widehat{f}^{(l+1)}([\mathbf{x}_n]) - f([\mathbf{x}_n]) = (\mathbf{I} - \mathbf{P}_{nn}^{(l)})(\widehat{f}^{(l)}([\mathbf{x}_n]) - f([\mathbf{x}_n]), \tag{5.3}$$

where $\mathbf{P}_{nn}^{(l)} := \mathbf{K}_{nn}^{(l)} (\mathbf{K}_{nn}^{(l)} + \lambda n \mathbf{I})^{-1}$, cf. Lemma C.1. It follows that

$$\|\widehat{f}^{(l+1)}([\mathbf{x}_n]) - f([\mathbf{x}_n])\| \leq \|\mathbf{I} - \mathbf{P}_{nn}^{(l)}\| \|\widehat{f}^{(l)}([\mathbf{x}_n]) - f([\mathbf{x}_n])\|. \tag{5.4}$$

Consider the SVD $\mathbf{K}_{nn}^{(l)} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^\top$ where $\mathbf{\Sigma} = \operatorname{diag}(\sigma_{l,i})$ and $\sigma_{l,n} \leq \cdots \leq \sigma_{l,1}$. We then have

$$\|\mathbf{I} - \mathbf{P}_{nn}^{(l)}\| = \left\|\mathbf{U}\operatorname{diag}\left(\frac{n\lambda}{n\lambda+\sigma_{l,i}}\right)\mathbf{U}^\top\right\| = \left\|\operatorname{diag}\left(\frac{n\lambda}{n\lambda+\sigma_{l,i}}\right)\right\| = \frac{n\lambda}{n\lambda+\sigma_{l,n}} := 1-\varepsilon(l), \tag{5.5}$$

and Thm. 5.6 follows recursively from Eq. (5.4) and Eq. (5.5). □

From Thm. 5.6 it follows that the LP estimator will converge as $l \to \infty$, since $\sigma_{l,n} > 0$ and therefore $1-\varepsilon(l) \in (0,1)$ for all l. In Thm. 5.7 we characterise how $\varepsilon(l)$ depends on the level l to give insight on the nature of this convergence.

Theorem 5.7. *The LP estimator $\widehat{f}^{(l)}$ from Eq. (2.7) converges with increasing level L to the training data $f(\mathbf{x}_i)$, cf. Thm. 5.6, with the rate $\prod_{l=0}^{L}(1-\varepsilon(l))$, where*

$$1-\varepsilon(l) \leq \left(1 + C_{1,D}2^{-Dl}\exp\left(-C_{2,D}4^{-l}\right)/n\lambda\right)^{-1}, \tag{5.6}$$

for

$$C_{1,D} = \frac{1}{2}(6\sqrt{2})^D\Gamma(D/2+1)^{\frac{D-1}{D+1}}\left(\frac{\pi}{9}\right)^{\frac{D}{D+1}}\left(\frac{r_0}{\delta}\right)^D \quad \text{and} \quad C_{2,D} = 1152\left(\frac{\pi\Gamma^2(D/2+1)}{9}\right)^{\frac{2}{D+1}}\left(\frac{r_0}{\delta}\right)^2,$$

where Γ is the gamma function.

Furthermore, for $l > \log_2(\sqrt{D/2}(r_0/\delta))$ we have the tighter bound

$$1-\varepsilon(l) < \left(1 + \left(1-2^{1+\frac{1}{\ln 2}(C_3 D - g(l))}\right)/n\lambda\right)^{-1}, \tag{5.7}$$

where $g(l) = 4^{l-\log_2 r_0/\delta}$ and $C_3 = (\ln(1+1/4) + 2\ln 2)$.

Proof. (Eq. (5.6)) We bound $1-\varepsilon(l) := n\lambda/(n\lambda+\sigma_{l,n})$ by bounding the smallest eigenvalue of the kernel matrix $[\mathbf{K}_{nn}^{(l)}]_{ij} = \Phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$, namely $\sigma_{l,n}$. To do so we assume that there exists a lower bound on the minimal distance between any two points $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$, defined as $\delta := \min_{i\neq j\in\mathcal{X}}\|\mathbf{x}_i - \mathbf{x}_j\| > 0$.

Consider the Gaussian $\Phi(\mathbf{x}) = \exp(-\beta\|\mathbf{x}\|_2^2)$, $\beta > 0$, with the Fourier transform $\widehat{\Phi}(\omega) = (\pi/\beta)^{D/2}\exp(-\|\omega\|_2^2/4\beta)$. From [60, Corollary 12.4] we have the bound

$$\sigma_{l,n} \geq C_D 2^D(2\beta)^{-D/2}\delta^{-D}\exp(-4M_D^2/(\delta^2\beta)),$$

where

$$M_D = 12\left(\frac{\pi\Gamma^2(D/2+1)}{9}\right)^{1/(D+1)} \quad \text{and} \quad C_D = \frac{1}{2\Gamma(D/2+1)}\left(\frac{M_D}{2^{3/2}}\right)^D.$$

With $\beta = (\sqrt{2}2^{-l}r_0)^{-2}$ and inserting for M_D and C_D we then have

$$\begin{aligned}\sigma_{l,n} &\geq C_D 2^D 2^{-Dl}\left(\tfrac{r_0}{\delta}\right)^D\exp\left(-(2\sqrt{2}M_D)^2(r_0/\delta)^2 4^{-l}\right)\\ &= \tfrac{1}{2}(6\sqrt{2})^D\Gamma(D/2+1)^{\frac{D-1}{D+1}}\left(\tfrac{\pi}{9}\right)^{\frac{D}{D+1}}\left(\tfrac{r_0}{\delta}\right)^D 2^{-Dl}\\ &\quad\cdot\exp\left(-1152\left(\tfrac{\pi\Gamma^2(D/2+1)}{9}\right)^{\frac{2}{D+1}}\left(\tfrac{r_0}{\delta}\right)^2 4^{-l}\right) := B(l),\end{aligned} \tag{5.8}$$

The bound in Eq. (5.6) follows from this result. □

Proof. (Eq. (5.7)) When the level l becomes sufficiently large, the kernel matrix $\mathbf{K}_{nn}^{(l)}$ becomes diagonally dominant, and we can therefore bound the eigenvalues using Garschgorins Theorem [61, Thm. 1.1], which gives

$$|\sigma_{l,i} - [\mathbf{K}_{nn}^{(l)}]_{jj}| = |\sigma_{l,i} - 1| < \sum_{\substack{q=1,\\ q\neq j}}^{n}|[\mathbf{K}_{nn}^{(l)}]_{jq}| \quad \text{for} \quad i,j \in [n]. \tag{5.9}$$

To find a more explicit bound, we analyze the sum on the right-hand side. Consider a family of annuli $\{R_t\}_{t=0}^{\infty}$ where $R_t = \mathcal{B}(\mathbf{x}_j, 2^{t+1}\delta)\backslash\mathcal{B}(\mathbf{x}_j, 2^t\delta)$. Inspired by [28], we can interpret the right hand side of Eq. (5.9) as a sum over $\{R_t\}_{t=0}^{\infty}$. The entries of $\mathbf{K}_{nn}^{(l)}$ are defined as

$$[\mathbf{K}_{nn}^{(l)}]_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2r_l^2}\right), \quad \forall i,j \in [n],$$

where $r_l = 2^{-l}r_0$ for $r_0 > 0$. It follows

$$\sum_{\substack{q=1,\\ q\neq j}}^{n}|[\mathbf{K}_{nn}^{(l)}]_{jq}| = \sum_{t=0}^{\infty}\sum_{\mathbf{x}_q\in R_t}k^{(l)}(\mathbf{x}_j,\mathbf{x}_q) \leq \sum_{t=0}^{\infty}\left(\tfrac{2^{t+2}\delta}{\delta}+1\right)^D\exp\left(-(2^t\delta 2^{-1/2}r_l^{-1})^2\right),$$

where in the first term on the right-hand side we bound the number of summands using Lemma 5.1, and in the second we use $\|\mathbf{x}_q - \mathbf{x}_j\| \geq 2^t\delta$ for $\mathbf{x}_q \in R_t$. Note now that for all $T \geq 1$ there exists $C_T > 0$ such that $\exp(-r^2) \leq C_T r^{-T}$ holds for all $r > 0$. Such a constant is given by the Lambert W function and satisfies $C_T = \left(\frac{T}{2\mathrm{e}}\right)^{T/2}$. Moreover, $2^{t+2} + 1 \leq 2^{t+2+\alpha}$, for $\alpha \geq \ln(1 + 1/4)/\ln(2)$. Thus,

$$\sum_{\substack{q=1,\\ q\neq j}}^{n} |[\mathbf{K}_{nn}^{(l)}]_{jq}| \leq C_T\left(\frac{r_l}{\delta}\right)^T 2^{(2+\alpha)D+T/2} \sum_{t=0}^{\infty} 2^{t(D-T)} \leq 2 \cdot 2^{D(2+\alpha)-T/2(1+1/\ln(2))} T^{T/2}\left(\frac{r_l}{\delta}\right)^T,$$

where in the last step we have used $\exp(1) \geq 2^{1+1/\ln(2)}$ along with $\sum_{t=0}^{\infty} 2^{t(D-T)} \leq 2$, which holds for $D - T < 0$. We let $r_l = r_0 2^{-l}$ and define $F(T) := 2^{-T/2(1+1/\ln(2))} T^{T/2} 2^{-lT}\left(\frac{r_0}{\delta}\right)^T$. From Lemma C.4 we have the minimum of $F(T)$, which together with the choice $\alpha = \ln(1 + 1/4)/\ln 2$, gives

$$\sigma_{l,n} > 1 - \sum_{\substack{q=1,\\ q\neq j}}^{n} \left|\left[\mathbf{K}_{nn}^{(l)}\right]_{jq}\right| \geq 1 - 2^{1+\frac{1}{\ln 2}((\ln(1+1/4)+2\ln 2)D - g(l))}, \quad g(l) = 4^{l-\log_2 r_0/\delta}.$$

By defining $C_3 = (\ln(1 + 1/4) + 2\ln 2)$ and using that $1 - \varepsilon(l) := n\lambda/(n\lambda + \sigma_{l,n})$ this leads to the bound in Eq. (5.7). We note that this bound holds for $T^* > D$ which means that $l > \log_2(\sqrt{D/2}r_0/\delta)$. □

We note that the bound in Eq. (5.6) underestimates the rate of convergence for lower levels but improves as the levels increase. Furthermore, Thm. 5.7 shows that the convergence rate increases with the level l. In fact, the bound in Eq. (5.6) can be simplified with an *a fortiori* bound of the same form, where $C_{1,D} = \frac{1}{2}\left(\frac{12.76}{2^{3/2}}\right)^D\left(\frac{D^D}{\Gamma(D/2+1)}\right)\left(\frac{r_0}{\delta}\right)^D$ and $C_{2,D} = (12.76\sqrt{2}D)^2(r_0/\delta)^2$, which ensures that $1 - \varepsilon(l)$ decreases monotonically for $l < \log_2(\sqrt{D/2}(r_0/\delta)) + \log_2(25.52\sqrt{2})$. see Remark C.2 and Corollary C.3.

On the other hand, when $l > \log_2(\sqrt{D/2}(r_0/\delta))$ the tighter bound from Eq. (5.7) ensures that $1 - \varepsilon(l)$ continues to decreases monotonically. Moreover, as $l \to \infty$ each new level reduces the residual error by $(1 + 1/n\lambda)^{-1}$. We can also observe that the convergence rate is reduced by the number of training points n, but this effect can be mitigated by reducing the regularization parameter λ. We also note that Thm. 5.6 and Thm. 5.7 are derived for a vector of numbers on the training data $\Gamma_n \subset \mathcal{X}$, without assumptions on the target function. In other words, the LP estimator can approximate the training data for any function $f : \Gamma_n \to \mathbb{R}$, to arbitrary precision, by including sufficiently many levels.

Corollary 5.8. *If the residual* $\mathbf{d}^{(l)} = (\widehat{f}^{(l)}([\mathbf{x}_n]) - f([\mathbf{x}_n]))$ *at level* l *only projects non-trivially onto the eigenvectors with eigenvalue* $\sigma_{l,n} \geq \sigma_{\text{cut-off}}$*, then we say the residual is spectrally band-limited with respect to the kernel. If the residual* $\mathbf{d}^{(l)}$ *is spectrally band-limited, then* $1 - \epsilon(l) < n\lambda/(n\lambda + \sigma_{\text{cut-off}})$.

Proof. Follows from Eq. (5.3)-(5.5) with $P_{nn}^{(l)} = P_{nn,k}^{(l)} + \left(P_{nn,k}^{(l)}\right)^{\perp}$, where $P_{nn,k}^{(l)}$ is the projection on the eigenvectors associated with the k largest eigenvalues and $\left(P_{nn,k}^{(l)}\right)^{\perp}(\widehat{f}^{(l)}([\mathbf{x}_n]) - f([\mathbf{x}_n])) = 0$. □

6. Experiments

This section presents comparative numerical experiments of the proposed estimator on three problems. In Section 6.1 we consider a one-dimensional regression problem, and in Section 6.2 we consider a dumbbell-shaped domain that consists of two 5-dimensional spheres connected by a 2-dimensional plane. Lastly, in Section 6.3, we forecast the trajectory of a double pendulum, which is a well-known chaotic system [25].

We compare StreaMRAK with FALKON [2] and an LP modification of KRR (LP-KRR). Both FALKON and LP-KRR rely on the standard Nyström sub-sampling [51,52]. Furthermore, FALKON does not rely on a multi-resolution scheme but uses instead a single bandwidth, found by cross-validation.

Throughout the experiments, we set the threshold for the number of sub-samples (landmarks) in StreaMRAK to be $10\sqrt{|Q_l|}$, where Q_l is the set of nodes at level l in the DCT. We note that to choose the sub-sample size, FALKON and LP-KRR require n to be known beforehand. For FALKON we let the number of Nyström landmarks be $10\sqrt{n}$, where n is the number of training samples. Meanwhile, for LP-KRR we sub-sample $\sqrt{n}$ Nyström landmarks, which are then used for all levels.

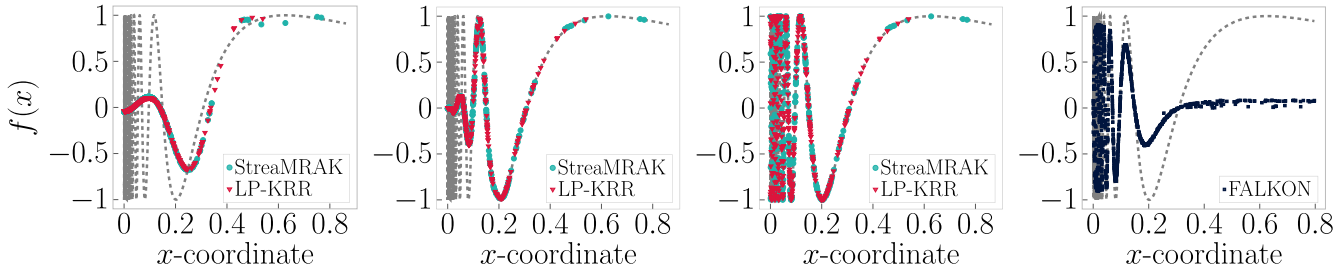
We also need to pre-select the number of training points for LP-KRR and FALKON. For FALKON we use the entire training set, as in [2]. Similarly, it is also common for the LP to use the entire training set at each level [27,28]. However, for large data sets, it might be better to include fewer data points. Therefore, we also use a version of the LP-KRR where we divide the total training data equally between the levels.

Throughout the experiments, we measure the performance of StreaMRAK, FALKON, and LP-KRR by estimating the mean square error

$$MSE(y, y_{pred}) = \frac{1}{\Upsilon\Lambda}\sum_{k=1}^{\Upsilon}\frac{1}{n_k}||\mathbf{y}_k - \mathbf{y}_k^{pred}||^2, \text{ with } \Lambda = \max_{\substack{k\in[\Upsilon]\\ i\in[n_k]}}[\mathbf{y}_k]_i - \min_{\substack{k\in[\Upsilon]\\ i\in[n_k]}}[\mathbf{y}_k]_i, \tag{6.1}$$

Table 1
Comparison of StreaMRAK, LP-KRR and FALKON for the target in Eq. (6.2). For each level l we show the number of landmarks, the mean square error (MSE), and the accumulated time to train the prediction model (Time). In parenthesis, in the time column of the FALKON row, is the time to find the optimal bandwidth through cross-validation.

	Level	# Landmarks	MSE	Time
	5	47	2.55×10^{-1}	77 s
StreaMRAK	10	392	3.69×10^{-2}	116 s
	15	1525	$8.63\times10-6$	497 s
	16	2302	$6.18\times10-6$	1194 s
	5	1483	2.56×10^{-1}	143 s
LP-KRR (1) $n_l = 1.1\times10^5$	10	1483	3.65×10^{-2}	413 s
	15	1483	$8.72\times10-6$	825 s
	16	1483	$6.85\times10-6$	922 s
	18	1483	$6.55\times10-6$	1136 s
	5	1483	2.56×10^{-1}	2963 s
LP-KRR (2) $n_l = 2.2\times10^6$	10	1483	3.64×10^{-2}	8704 s
	18	1483	$8.91\times10-6$	23113 s
StreaMRAK	–	14830	5.7×10^{-3}	4642 s+(27930 s)



(a) Level 3 (b) Level 5 (c) Level 16 (d) Single level

Fig. 3. (a)-(d) shows the target function $f(x)$ from Eq. (6.2) as a grey dotted line. The light-blue circles indicates the predicted values made by StreaMRAK. Similarly the red triangles indicates the predictions made by LP-KRR and the dark blue squares the predictions made by FALKON. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

where Υ is the number of test runs we average over, n_k is the number of test points at test run k, and $\mathbf{y}_k, \mathbf{y}_k^{pred} \in \mathbb{R}^{n_t}$ are the target values and predictions respectively, and Λ is the normalisation factor.

6.1. Multi-resolution benchmark

We consider the function,

$$f(x) = \sin\left(\frac{1}{x+0.01}\right), \quad \text{for } x \in \left[0, \frac{\pi}{4}\right]. \tag{6.2}$$

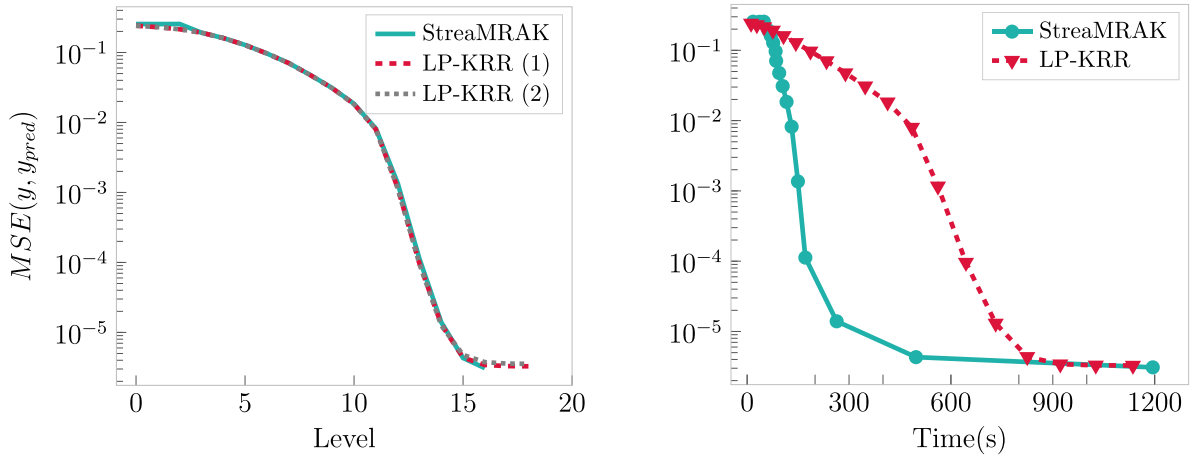
In the experiment we use a training set of $n = 2.2\times10^6$ samples and a test set of 1.3×10^5 samples. We use the non-uniform gamma distribution $\Gamma(\alpha, \beta)$ with $\alpha = 1$, $\beta = 2$ to sample the training data.

The number of training points used at each level in StreaMRAK is determined by setting δ_1 and δ_2 from Def. 4.1 to 10^{-3}. With this choice, StreaMRAK selects between 30244 and 40100 training points for each level. For comparison, FALKON uses all the 2.2×10^6 training points. Furthermore, for LP-KRR we run two experiments: LP-KRR (1) using 1.1×10^5 training points at each level and LP-KRR (2) using 2.2×10^6 training points at each level.

Results are presented in Table 1, and the prediction results are illustrated in Fig. 3a-3 d. The results show that StreaMRAK and both LP-KRR schemes perform much better than FALKON. The reason is that FALKON uses only one bandwidth r, while the multi-resolution schemes StreaMRAK and LP-KRR, utilize a bandwidth regime $r_l = 2^{-l}r_0$ that varies with the level l. The consequence is that StreaMRAK and LP-KRR approximate the low-frequency components of f when the bandwidth is large, and then target the high-frequency components of $f(x)$ gradually as the bandwidth decreases. These results illustrate the benefits of a multi-resolution scheme over a single bandwidth scheme.

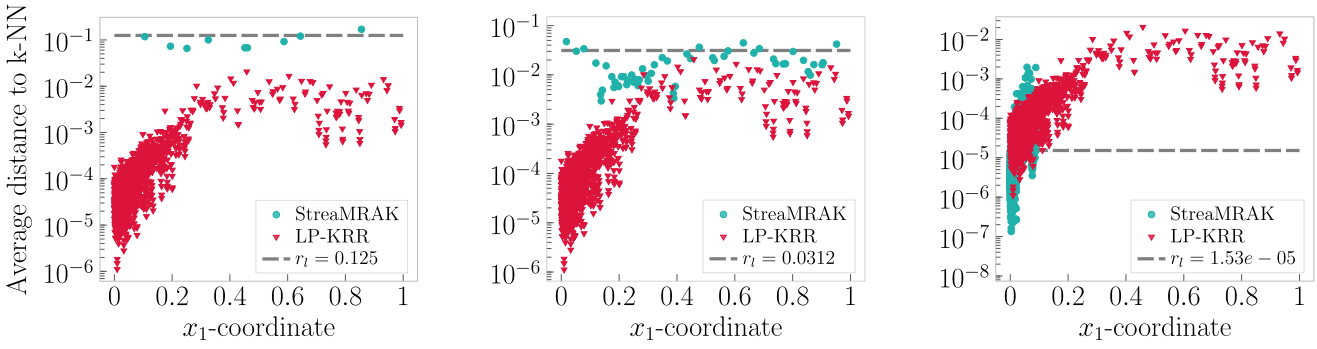
From Table 1, we also observe that LP-KRR (2) is significantly slower than StreaMRAK and LP-KRR (1). This is because it uses the entire training set at each level. Therefore, since LP-KRR (1) and LP-KRR (2) achieve comparable precision, we see that including all training points at each level is not always necessary.

A closer comparison of StreaMRAK and LP-KRR is given in Fig. 4. In particular, in Fig. 4a we see that the two algorithms achieve very similar precision. However, comparing the training times in Fig. 4b, we see that StreaMRAK trains each level faster and therefore achieves better precision earlier than LP-KRR (1).



(a) Mean square error (b) Training time

Fig. 4. Comparison of StreaMRAK and LP-KRR. (a) shows the mean square error calculated according to Eq. (6.1) with the target function from Eq. (6.2). Along the x-axis is the number of levels included in the model. (b) The x-axis shows the accumulated training time until a level in the LP is completed. The y-axis shows the MSE of the prediction using the currently available model. The blue circles indicate the prediction error of StreaMRAK and the red triangles indicate the prediction error of LP-KRR (1). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Level 3 (b) Level 5 (c) Level 16

Fig. 5. (a)-(c) shows the landmarks with their position along the x_1 axis and the average distance to their 2 nearest neighbors along the y-axis. Here the red triangles are the Nyström landmarks of LP-KRR and the light-blue circles the landmarks of StreaMRAK. The grey dotted line is the bandwidth at the given level. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In Fig. 5 we show the average distance of each landmark to their 2 nearest neighbors (2-NN distance). Two aspects of the selection require attention. As opposed to LP-KRR, StreaMRAK selects landmarks such that the 2-NN distance is comparable to the bandwidth used at a specific level. In addition, StreaMRAK saves computational power by not choosing landmarks in regions where the 2-NN distance is too low compared to the bandwidth. In Fig. 5c this can be observed for level $l = 16$ for landmarks with $x \geq 0.2$. Due to the non-uniform sample distribution with a higher density around $x = 0$, the adaptive sub-sampling is able to select more landmarks in the region close to $x = 0$, where f oscillates with high frequency. Furthermore, StreaMRAK stops predicting at level 16 because level 17 is not yet covered with a high enough density of landmarks. Meanwhile, LP-KRR continues, but as seen from Fig. 4a the improvements after level 15 are not significant because the density of Nyström samples is too low compared to the bandwidth.

6.2. Adaptive sub-sampling benchmark

We consider a dumbbell-shaped domain embedded in $\mathbb{R}^5$, consisting of two 5-dimensional spheres connected by a 2-dimensional plane. A projection of the input domain in $\mathbb{R}^3$ is shown in Fig. 7 (a)-(c). Furthermore, as target we consider the following function,

$$f(\mathbf{x}) = \begin{cases} A\sin(Bx_1 + \phi) + (x_1 + 2), & 1 < x_1 < 3 \\ 1, & \text{otherwise} \end{cases}, \quad \text{for } \mathbf{x} \in [-1, 5] \times [-1, 1]^4, \tag{6.3}$$

where A, B and ϕ are chosen so that $f \in C^1([-1, 5] \times [-1, 1]^4, \mathbb{R}^5)$. For the experiments, we consider a training set of 1.9×10^6 samples and a test set of 6×10^5 samples, all sampled uniformly at random from the input domain. We note that

Table 2
Comparison of StreaMRAK, LP-KRR, and FALKON predictions of the target function in Eq. (6.3). For each level l we show the number of landmarks, the mean square error (MSE), and the accumulated time to train the prediction model (Time). In parenthesis, in the time column of the FALKON row, is the time to find the optimal bandwidth through cross-validation.

	Level	# Landmarks	MSE	Time
	4	352	1.29×10^{-3}	64 s
StreaMRAK	5	2667	1.27×10^{-3}	1398 s
	6	1858	8.31×10^{-4}	1462 s
	8	1329	2.75×10^{-5}	2307 s
	4	1375	1.28×10^{-3}	386 s
LP-KRR (1) $n_l = 1.8\times10^5$	5	1375	1.26×10^{-3}	520 s
	6	1375	9.10×10^{-4}	671 s
	8	1375	3.30×10^{-4}	1064 s
	9	1375	3.10×10^{-4}	1287 s
	4	1375	1.34×10^{-3}	4160 s
LP-KRR (2) $n_l = 1.9\times10^6$	5	1375	1.30×10^{-3}	5570 s
	6	1375	9.44×10^{-4}	7168 s
	8	1375	3.16×10^{-4}	11125 s
	9	1375	3.01×10^{-4}	13334 s
FALKON	–	14830	6.8×10^{-4}	6590 s+(37561 s)

we purposefully chose a simple function in the high dimensional regions because complicated functions in high dimensions require far too many points to be satisfactorily learned.

To determine the number of training points for StreaMRAK, we let $\delta_1 = 1\times10^{-3}$ and $\delta_2 = 1\times10^{-4}$, cf. Def. 4.1. With this choice StreaMRAK selects between 30100 and 40100 training points for each level. FALKON again uses all the 1.9×10^6 training points and for LP-KRR we consider two settings: LP-KRR (1) using 1.8×10^5 training points at each level, and LP-KRR (2) using 1.9×10^6 training points at each level.

The results for StreaMRAK, LP-KRR, and FALKON are presented in Table 2. We observe that StreaMRAK achieves a better prediction than both FALKON and LP-KRR because it adapts the sub-sampling density to the level of resolution.

To understand the improvement in prediction accuracy, we need to discuss the effects of landmark selection. In Fig. 7a-7c we show the projections of landmarks for StreaMRAK and LP-KRR on $\mathbb{R}^3$, and in Fig. 7d-7f the average distance of each landmark to its 7 nearest neighbors. These distances are compared with the bandwidth r_l selected for the given level l. We see that StreaMRAK selects landmarks in regions where the average distance to nearest neighbors is comparable to the bandwidth. This means that in high dimensional regions, which correspond to $x_1 \in [-1,1]\cup[3,5]$, the algorithm effectively stops collecting landmarks since it cannot maintain high enough density. On the other hand, LP-KRR uses Nystrom sub-sampling, which imposes a uniform selection of landmarks. Consequently, a significant number of landmarks come from high-dimensional regions.

Moreover, Fig. 7 shows that in the case of LP-KRR, the average distance between the landmarks in high dimensional regions is larger than the bandwidth r_l when $l \geq 5$. As a knock-on effect, LP-KRR makes only small improvements in high dimensional regions for $l \geq 5$, as seen from Fig. 6b. Analogous behavior can be observed for StreaMRAK. However, since StreaMRAK devotes fewer resources to high dimensional regions, it sub-samples more from the low dimensional region, as illustrated in Fig. 6a. The consequence is that StreaMRAK makes bigger improvements in the low dimensional region than LP-KRR, as seen from Fig. 6b. Note that this was not the case in Section 6.1, where the two methods had similar behavior, but unlike here, the input domain in Section 6.1 did not consist of regions with different dimensionalities.

6.3. Forecasting the trajectory of a double pendulum

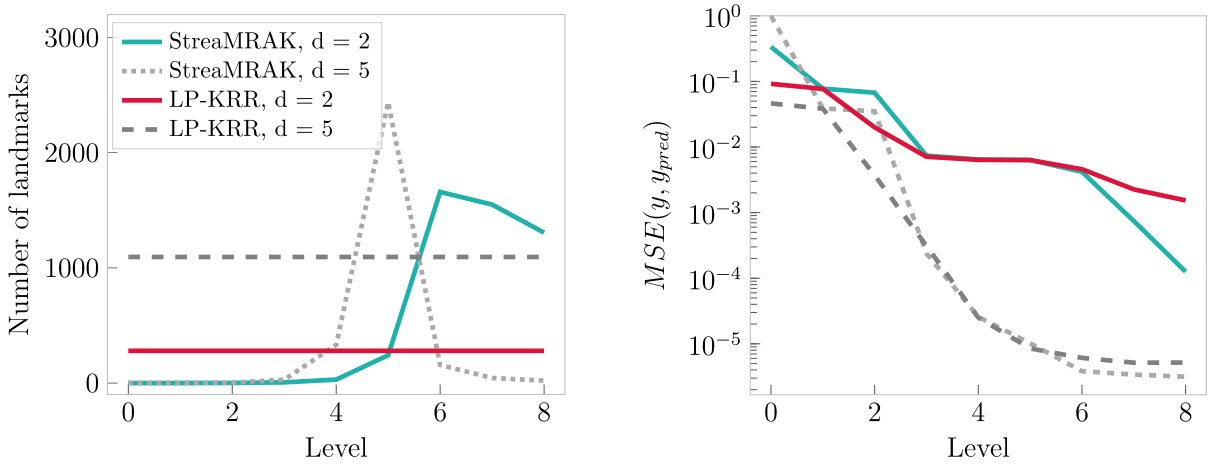
We consider the double pendulum, illustrated in Fig. 2a, which we model by the Lagrangian system

$$\mathcal{L} = ml^2(\omega_1^2 + \frac{1}{2}\omega_2^2) + ml^2\omega_1\omega_2\cos(\theta_1-\theta_2) + mgl(2\cos\theta_1 + \cos\theta_2), \tag{6.4}$$

under the assumption that the pendulums are massless rods of length $l_1 = l_2 = l$ with masses $m_1 = m_2 = m$ centered at the end of each rod. Here g is the standard gravity, $\omega_1 := \dot{\theta}_1$, $\omega_2 := \dot{\theta}_2$ are the angular velocities, and the angles θ_1, θ_2 are as indicated in Fig. 2a. For the experiments we let $m = 1$, $l = 1$ and $g = 10$.

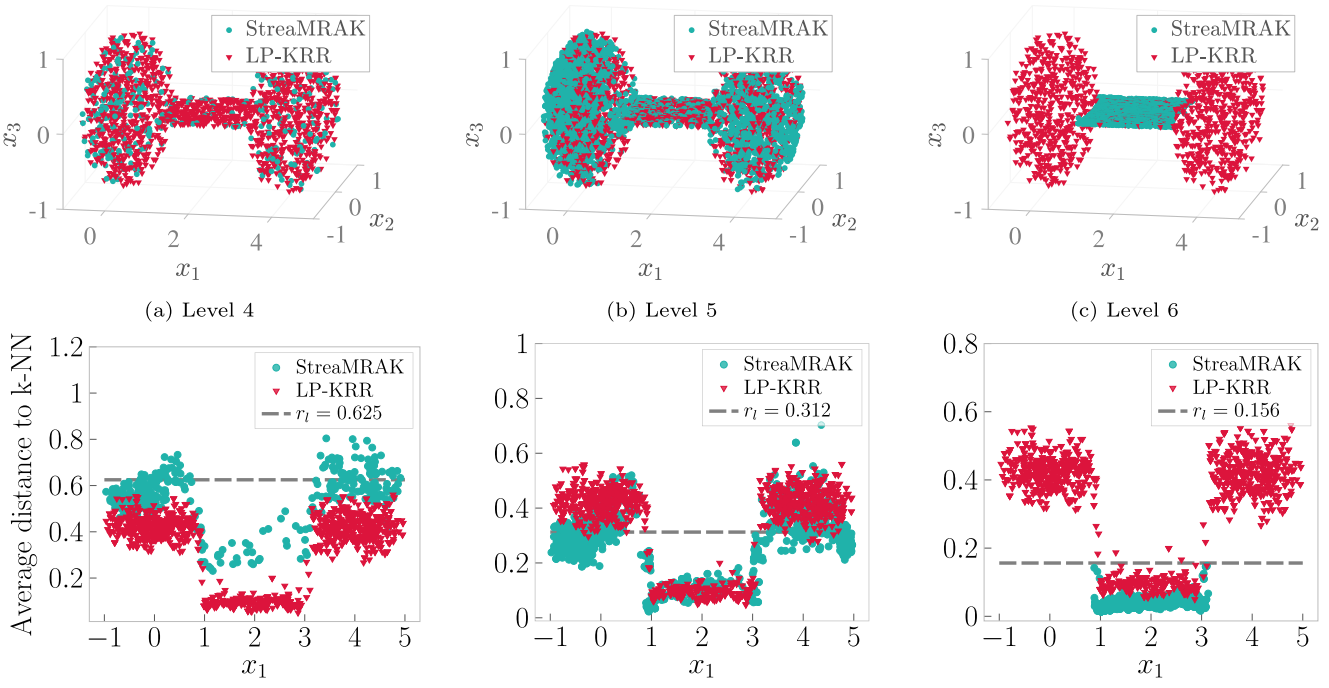
The learning task is to forecast the trajectory of the pendulum, given only its initial conditions. We let $\mathbf{s}_t = [\theta_1(t), \theta_2(t), \omega_1(t), \omega_2(t)] \in \mathbb{R}^4$ be the state of the system at step $t \in \mathbb{N}$ and train StreaMRAK, LP-KRR and FALKON to learn how $\mathbf{s}_t$ maps to a later state $\mathbf{s}_{t+\Delta}$, for $\Delta \in \mathbb{N}$. The trained model $\widehat{f}$ is used to forecast the state $\mathbf{s}_T$ for $T >> 0$ by recursively predicting $\mathbf{s}_{t+\Delta} = \widehat{f}(\mathbf{s}_t)$ from the initial state $\mathbf{s}_0$ until $t = T$.

For the experiments we consider two settings: a low energy system $\mathbf{s}_0^{low} = [-20^\circ, -20^\circ, 0^\circ, 0^\circ]$ and a high energy system $\mathbf{s}_0^{high} = [-120^\circ, -20^\circ, -7.57^\circ, 7.68^\circ]$. For these systems, we initialize 8000 pendulums as $\mathbf{s}_0 \sim \mathcal{N}(\mathbf{s}, \sigma(\mathbf{s}))$ for $\mathbf{s} = \mathbf{s}_0^{low}, \mathbf{s}_0^{high}$



(a) Landmarks distribution (b) Mean square error

Fig. 6. Comparison of StreaMRAK and LP-KRR (1) in the 2-dim and 5-dim regions of the Dumbbell domain. The solid blue line is StreaMRAK for dimension $d = 2$ while the solid red line is LP-KRR (1) for dimension $d = 2$. The grey dotted line is StreaMRAK for dimension $d = 5$ and the dark-grey dashed line is LP-KRR (1) for dimension $d = 5$ (a) shows the mean square error calculated according to Eq. (6.1). (b) shows the number of landmarks in the 2-dimensional and the 5- dimensional regions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Level 4 (b) Level 5 (c) Level 6

(d) Level 4 (e) Level 5 (f) Level 6

Fig. 7. In the figure, red triangles correspond to LP-KRR and light-blue circles to StreaMRAK. (a)-(c) shows the landmark distributions projected on $\mathbb{R}^3$ at level $l = 4, 5, 6$ respectively. (d)-(f) shows the average distance between the 7 nearest neighbors; bandwidth r_l is indicated with a dotted line. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

respectively, where $\sigma(\mathbf{s}) = [0.025|\theta_1|,\ 0.15|\theta_2|,\ 0.3|\omega_1|,\ 0.3|\omega_2|]$. Each pendulum is iterated for 500 steps, which results in 5×10^6 training points distributed in $\mathbb{R}^4$. Furthermore, for the test data we consider 100 pendulums $\mathbf{s}_0 \sim \mathcal{N}(\mathbf{s}, 0.01|\mathbf{s}|)$ for $\mathbf{s} = \mathbf{s}_0^{low}, \mathbf{s}_0^{high}$, iterated for 500 steps.

To determine the number of training points for StreaMRAK, we let $\delta_1, \delta_2 = 10^{-4}$, cf. Def. 4.1. With this choice StreaMRAK selects between 30219 and 70282 training points for each level for the low energy system, and between 36300 and 130200 for the high energy system. Meanwhile, FALKON uses all 5.0×10^6 training points and LP-KRR use 3.9×10^5 training points at each level.

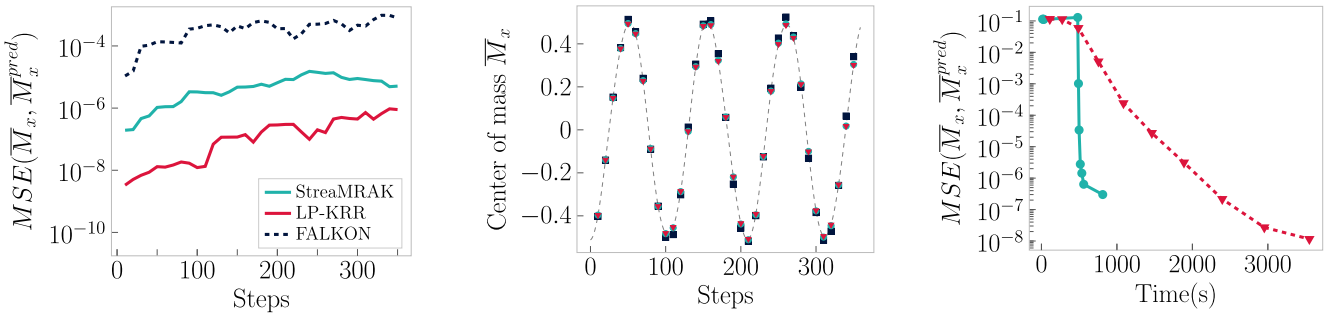
Results are presented in Table 3 and Table 4. Furthermore, to illustrate the prediction results we consider the center of mass $\overline{M}_x(\mathbf{s}_t) = \frac{1}{2}(x_1(t) + x_2(t)) \in \mathbb{R}$ at state $\mathbf{s}_t$, where $x_1, x_2 \in \mathbb{R}$ are the positions of the two pendulum masses as seen in

Table 3
Comparison of StreaMRAK, LP-KRR, and FALKON for the low energy system. For each level l we show the number of landmarks, the MSE at step T=50, and the accumulated time to train the prediction model (Time). In parenthesis, in the time column of the FALKON row, is the time to find the optimal bandwidth through cross-validation.

	Level	# Landmarks	MSE(T=50)	Time
StreaMRAK	2	1	1.12×10^{-1}	31 s
	5	66	3.39×10^{-5}	498 s
	7	659	1.44×10−6	534 s
	9	4085	3.01×10^{-7}	812 s
LP-KRR	2	1979	5.93×10^{-2}	490 s
	5	1979	2.73×10^{-5}	1463 s
	7	1979	2.16×10^{-7}	2395 s
	9	1979	1.16×10^{-8}	3550 s
FALKON	–	19790	5×10^{6}	2934 s+(1498 s)

Table 4
Comparison of the StreaMRAK, LP-KRR, and FALKON for the high energy system. For each level l we show the number of landmarks, the MSE at step T=50, and the accumulated time to train the prediction model (Time). In parenthesis, in the time column of the FALKON row, is the time to find the optimal bandwidth through cross-validation.

	Level	# Landmarks	MSE(T=50)	Time
StreaMRAK	2	1	2.70×10^{-1}	49 s
	5	1106	8.53×10^{-3}	915 s
	7	6376	2.16×10^{-4}	1999 s
LP-KRR	2	1979	1.72×10^{-2}	522 s
	5	1979	5.09×10^{-3}	1474 s
	7	1979	1.39×10^{-4}	2431 s
FALKON	–	19790	5×10^{6}	23830 s+(11050 s)



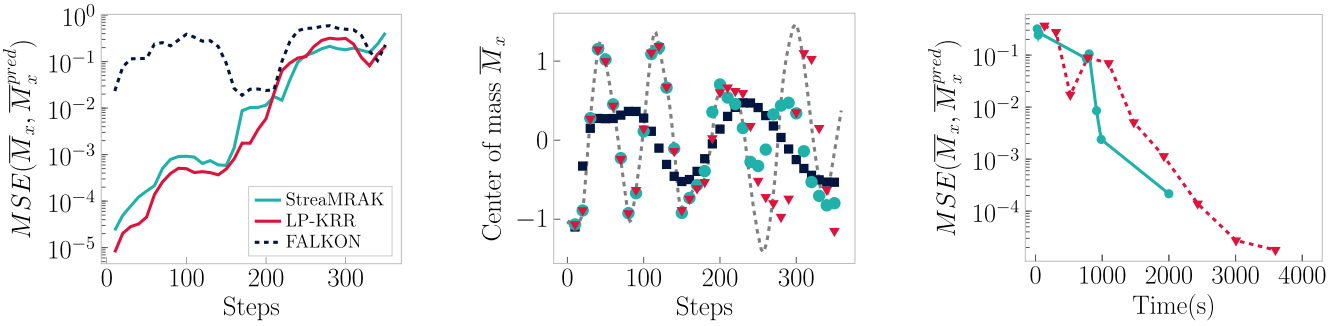
(a) Mean square error (b) Forecasting trajectory (c) Training time

Fig. 8. Comparison of StreaMRAK (light blue lines and circles), LP-KRR (red lines and triangles), and FALKON (dark blue dotted lines and squares) for the low energy pendulum. (a) Shows the mean square error of the center of mass $\overline{M}_x(\mathbf{s}_t)$ for the level 7 prediction, with step T along the x-axis. (b) shows the true center of mass trajectory as a grey dotted line and the predictions of StreaMRAK, LP-KRR, and FALKON at level 9. (c) The x-axis shows the accumulated training time until a level in the LP is completed. The y-axis shows the MSE of the predicted system state after $T = 50$ steps. We note that StreaMRAK includes 7 levels, while LP-KRR includes 9. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 2a. The prediction results are illustrated in Fig. 8 and Fig 9 for the low and high energy pendulums respectively. We calculate the MSE at each step t separately, such that for a given t we use Eq. 6.1 with $\mathbf{y}_k = \overline{M}_x(\mathbf{s}_t)$, $\mathbf{y}_k^{pred} = \overline{M}_x(\mathbf{s}_t^{pred})$ and $\Upsilon = 100$.

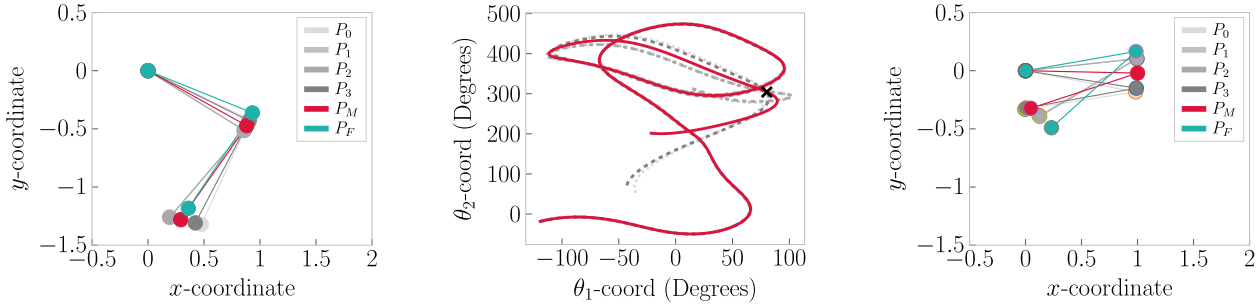
For the low energy system, we see from Fig. 8c how StreaMRAK is trained significantly faster than LP-KRR, although at a cost of reduced precision. The reduced training time of StreaMRAK is a consequence of the low doubling dimension of the training data, which allows the selection of far fewer landmarks for StreaMRAK than what is used at each level in LP-KRR.

For the high-energy pendulum, we see from Fig. 9c that StreaMRAK is again able to achieve good precision faster than LP-KRR. Furthermore, we see that the number of landmarks selected for StreaMRAK increases abruptly with the levels, reflecting the high doubling dimension of the training data. Due to this StreaMRAK stops the training after level 7, as



(a) Mean square error (b) Forecasting trajectory (c) Training time

Fig. 9. Comparison of StreaMRAK (light blue lines and circles), LP-KRR (red lines and triangles), and FALKON (dark blue dotted lines and squares) for the high-energy pendulum. (a) Shows the mean square error of the center of mass $\overline{M}_{x(\mathbf{s}_t)}$ for the level 7 prediction, with step T along the x-axis. (b) shows the true center of mass trajectory as a grey dotted line and the predictions of StreaMRAK, LP-KRR, and FALKON at level 9. (c) The x-axis shows the accumulated training time until a level in the LP is completed. The y-axis shows the MSE of the predicted system state after $T = 50$ steps. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Step $T = 200$. (b) Projection of the pendulum trajectories on the $\theta_1\theta_2$-plane. (c) Step $T = 210$.

Fig. 10. (a) Pendulum positions at $T = 200$ and (c) The positions at $T = 210$. In (a) and (c), P_M is the main pendulum with initial conditions $\mathbf{s}_0^{high}$, while P_F is the StreaMRAK forecast of the pendulum position. Similarly, $P_0 - P_3$ are four training pendulums with a perturbation of 0.5% on the initial angles θ_1 and θ_2 of the main pendulum. (b) Projection of the training data on the $\theta_1\theta_2$-plane. The thick red line is the main pendulum corresponding to P_M and the four grey dotted lines are the test pendulums $P_0 - P_3$, where the X indicates the time $T = 205$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the next levels require too many landmarks. By continuing for 2 more levels LP-KRR is able to achieve marginally better precision but at increased computational cost.

As seen in Fig. 9b, the forecasting of StreaMRAK and LP-KRR breaks down after $T \approx 200$ steps. In Fig. 10b we observe the trajectory of a pendulum with initial condition $\mathbf{s}_0^{high}$, as well as four pendulums with a 0.5% perturbation on the angles θ_1 and θ_2 in $\mathbf{s}_0^{high}$. We observe that after roughly $T = 205$ time steps the trajectory of the five pendulums diverge significantly from each other. Therefore, it seems that a bifurcation point occurs around this time, which may explain why all the algorithms are unable to make good forecasting beyond this point.

7. Outlook

Further development of StreaMRAK is intended with focus on four objectives.

(O1) Augmentation of the DCT to track the error at each node
(O2) Improve the estimator in Def. 4.1 and Eq. 3.1.
(O3) Refinement of previously fitted levels in the LP as new data arrives.
(O4) Further theoretical analysis of the LP.

Considering objective (O1) we intend to develop the DCT to track the error at each node. This way the growth can be restricted in regions where the error is small, which allows for more focus on regions where the error is large. The intention is that this will reduce the problem complexity even further, while also increasing the precision. Regarding objective (O2), a drawback with the estimator in Eq. 3.1 was already mentioned in Remark B.1 in Appendix B. Furthermore, for the estimator in Def. 4.1, we intend to implement and evaluate alternative ways to estimate the convergence of the matrices. Another focus area will be objective (O3), as we believe new information may be revealed as new training data arrive, and refinement of previously fitted levels can therefore be beneficial. Finally, the theoretical analysis in objective (O4) will focus on analyzing the generalization error for the LP, particularly in combination with the adaptive sub-sampling scheme.

Acknowledgement

We especially would like to thank Prof. Pieter Abeel at UC Berkeley and Asst. Prof. Sicun Gao at UC San Diego for their input on the double pendulum system, and for providing a code example for this system. We would also like to thank Sami Ortoleva at UC San Diego for his discussion on the analysis of the damped cover-tree. AO is part of the Simula-UCSD-UiO Research and Ph.D. training program (SUURPh), an international collaboration in computational biology and medicine funded by the Norwegian Ministry of Education and Research, ŽK is funded by UK EPSRC grant EP/T000864/1, AC is funded by NSF DMS 1819222, 2012266, and Russell Sage Foundation grant 2196 and YF is funded by the NIH grant NINDS (PHS) U19NS107466 Reverse Engineering the Brain Stem Circuits that Govern Exploratory Behavior.

Appendix A. Algorithms

We here denote nodes by p, q, c and $\mathbf{x}_p,\ \mathbf{x}_q, \mathbf{x}_c \in \mathcal{X} \subset \mathbb{R}^D$ are the corresponding points.

Algorithm A.1 INSERT(point q, node p, level l).

We assume q already satisfies $\|\mathbf{x}_q - \mathbf{x}_p\| \leq 2^{-l} r_0$.
if $\|\mathbf{x}_q - \mathbf{x}_c\| > 2^{-(l+1)} r_0$ for all $c \in \mathit{Children}(p)$ **then**
  Insert q into $\mathit{Children}(c)$.
  UPDATE_COVERFRACTION($\mathit{Parent}(Q_l)$, "No parent found")
  Break
else if $\|\mathbf{x}_q - \mathbf{x}_c\| < 2^{-(l+1)} r_0$ for some $c \in \mathit{Children}(p)$ **then**
  Consider all children of c, namely $\mathit{Children}(c)$
  if $\mathit{Children}(c)$ is empty **then**
    if Covering fraction of p, Def. 3.1, satisfy $\mathfrak{cf}(p) \geq \mathcal{D}_{\mathfrak{cf}}$ for some threshold $\mathcal{D}_{\mathfrak{cf}}$ **then**
      Insert q into $\mathit{Children}(c)$
      Break
    else
      UPDATE_COVERFRACTION(p, "parent found"){c is found to be a potential parent. However, since $\mathfrak{cf}(p) < \mathcal{D}_{\mathfrak{cf}}$ we can not add q to $\mathit{Children}(c)$}
    end if
  else
    INSERT(q, c, $l+1$)
  end if
end if

Algorithm A.2 STREAMRAK(point $\mathbf{x}$, target y).

Let l be the level. Let $p_{(0)}$ be the root node, r_0 the radius of the root node.
Sub-sampling thread
Insert $\mathbf{x}$ into the cover tree with INSERT($\mathbf{x}, p_{(0)}, l = 0$). {See Alg. A.1}
if a new level has $\mathfrak{cf}(Q_l) \geq \mathcal{D}_{level}$. **then**
  Extract the landmarks at level l as sub-samples, namely $\Gamma^{(l)}_{m^{(l)}}$.
end if
Training thread
Consider level l and assume that the landmarks $\Gamma^{(l)}_{m^{(l)}}$ are extracted.
while l is not sufficiently covered with training points according to Def. 4.1. **do**
  Update $\left[(\mathbf{K}^{(l)}_{nm})^\top \mathbf{K}^{(l)}_{nm}\right]_{ij}$ and $\mathbf{z}^{(l)}_i$ according to Eq. 2.4 and Eq. 2.5 as new samples $(\mathbf{x}, y)$ arrive, using the landmarks in $\widetilde{\Gamma}^{(l)}_m$ from Def. 3.3.
  Continuously check if matrices have converged.
  if Matrices converge according to Def. 4.1 **then**
    Update the StreaMRAK regression model $\widetilde{f}^{(L)}$, by including the correction term $s^{(l)}$ into the Laplacian pyramid, as described in Section 2.2. Let $L = l$ and update $l = l + 1$.
  end if
end while

Algorithm A.3 UPDATE_COVERFRACTION(node p, string s).

if s="No parent found" **then**
    Update covering fraction of p with $\mathfrak{cf}(p) = (1-\alpha)\mathfrak{cf}(p)$
else if s= "parent found" **then**
    Update covering fraction of p with $\mathfrak{cf}(p) = (1-\alpha)\mathfrak{cf}(p) + \alpha$
end if

Appendix B. Preparatory material

We offer preparatory material on the damped cover-tree and kernel methods.

B1. Preparatory material on the damped cover-tree

This section shows how the recursive formula in Eq. 3.1 approximates the weighted average of the outcome of the last N random trails. Where the trails are as described in Section 3.1. By expanding Eq. 3.1 we have $(\mathfrak{cf}(p))_t = (1-\alpha)^t(\mathfrak{cf}(p))_1 + \alpha\sum_{i=1}^{t-1}(1-\alpha)^i\mathbb{1}_{\mathcal{B}_c}(\mathbf{x}_{t-i})$. Since $(1-\frac{1}{N})^N \approx 1/e$, the first term becomes negligible when $t \gg N$. Similarly, all terms $i > N$ in the sum becomes negligible. This leaves,

$$(\mathfrak{cf}(p))_t \approx \frac{1}{N}\sum_{i=1}^{N}\left(1-\frac{1}{N}\right)^i\mathbb{1}_{\mathcal{B}_c}(\mathbf{x}_{t-i})$$

which is a weighted average of the outcome of the N last draws as claimed.

Remark B.1. We mention a weakness of the estimator in Eq. (3.1). As follows from Algorithm A.1, every time a new point $\mathbf{x}$ is not covered by the existing children, a new child is added. This consequently updates $\mathcal{B}_c$, leading to the posterior distribution $\text{Prob}(\mathbb{1}_{\mathcal{B}_c}(\mathbf{x}) = 0|\mathbf{x})$ to changed every time $\mathbb{1}_{\mathcal{B}_c}(\mathbf{x}) = 0$.

B2. Preparatory material on Kernel methods

Kernel methods in the context of reproducing kernel Hilbert spaces (RKHS) offer a powerful approach to machine learning with a well-established mathematical foundation [1,62]. In this paper we consider an input space $\mathcal{X} \subset \mathbb{R}^D$, a corresponding target space $\mathcal{Y} \subset \mathbb{R}$ and let ρ be the probability distribution on $\mathcal{X}\times\mathcal{Y}$. Furthermore, we assume an RKHS $\mathcal{H}_k$ generated by a positive definite kernel $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$. In other words, the eigenvalues $\sigma_i, \dots, \sigma_n$ of the corresponding kernel matrix $\mathbf{K}_{nn} = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^n$ satisfies $\sigma_i > 0$ for all $i \in n$. In this setting the inner product between two feature vectors $\phi(\mathbf{x}), \phi(\mathbf{x}') \in \mathcal{H}_k$ satisfies the property that $< \phi(\mathbf{x}), \phi(\mathbf{x}') >_{\mathcal{H}_k} = k(\mathbf{x}, \mathbf{x}')$. This relation, known as the "kernel trick" [63,64], effectively circumvents the need for explicit construction of non-linear mappings ϕ.

Given a training set $\{(\mathbf{x}_i, y_i) : i \in [n]\}$ sampled according to ρ with $\Gamma_n = \{\mathbf{x}_i : i \in [n]\}$, we formulate the kernel ridge regression (KRR) problem as

$$\widehat{f}_{n,\lambda} = \underset{f\in\widehat{\mathcal{H}}_n}{\text{argmin}}\ \frac{1}{n}\sum_{i=1}^{n}(f(\mathbf{x}_i) - y_i)^2 + \lambda\|f\|_{\mathcal{H}}^2, \tag{B.1}$$

where $\lambda > 0$ is a regularisation parameter and $\widehat{\mathcal{H}}_n = \overline{\text{span}}\{k(\cdot, \mathbf{x}_i) : i \in [n]\}$ is a finite-dimensional subspace of $\mathcal{H}_k$. What is more, for all $f \in \widehat{\mathcal{H}}_n$ the Representer theorem [65,66] guarantees that there exists coefficients $\alpha_1, \dots, \alpha_n$ such that the solution to Eq. (B.1) is on the form

$$f(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i k(\mathbf{x}, \mathbf{x}_i).$$

Computing the KRR estimator is therefore reduced to solving the linear system

$$(\mathbf{K}_{nn} + \lambda\mathbf{I}_n)\boldsymbol{\alpha} = \mathbf{y},$$

where $\mathbf{y} = (y_1, \dots, y_n)^\top$, $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n)^\top$, and $[\mathbf{K}_{nn}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

Appendix C. Auxiliary results

Lemma C.1. *Let $\mathbf{d}^{(l)}$ be the residual at level l as defined in Eq. (2.9). We then have,*

$$\mathbf{d}^{(l+1)} = (\mathbf{I} - \mathbf{K}_{nn}^{(l)}(\mathbf{K}_{nn}^{(l)} + \lambda n\mathbf{I})^{-1})\mathbf{d}^{(l)}$$

Proof. Denote $\mathbf{s}^{(l)} = s^{(l)}([\mathbf{x}_n])$, and note that $\mathbf{s}^{(l)} = \mathbf{K}_{nn}^{(l)}(\mathbf{K}_{nn}^{(l)} + \lambda n\mathbf{I})^{-1}\mathbf{d}^{(l)}$. For $l = 1$, we have

$$\mathbf{d}^{(1)} = \mathbf{y} - \mathbf{s}^{(0)} = \mathbf{y} - \mathbf{K}_{nn}^{(l)}\boldsymbol{\alpha}^{(0)} = (\mathbf{I} - \mathbf{K}_{nn}^{(0)})(\mathbf{K}_{nn}^{(0)} + \lambda n\mathbf{I})^{-1}\mathbf{y}.$$

We proceed by induction. Assume the statement holds for an $l \geq 2$. We now have

$$\mathbf{d}^{(l+1)} = \mathbf{y} - \sum_{j=0}^{l}\mathbf{s}^{(j)} = \mathbf{d}^{(l)} - \mathbf{s}^{(l)} = \mathbf{d}^{(l)} - \mathbf{K}_{nn}^{(l)}(\mathbf{K}_{nn}^{(l)} + \lambda n\mathbf{I})^{-1}\mathbf{d}^{(l)} = (\mathbf{I} - \mathbf{K}_{nn}^{(l)}(\mathbf{K}_{nn}^{(l)} + \lambda n\mathbf{I})^{-1})\mathbf{d}^{(l)}.$$

□

Remark C.2. In [60, Thm. 12.3] they also offer an a fortiori bound corresponding to $M_D = 6.38D$, $C_{1,D} = \frac{1}{2}\left(\frac{12.76}{2^{3/2}}\right)^D\left(\frac{D^D}{\Gamma(D/2+1)}\right)\left(\frac{r_0}{\delta}\right)^D$ and $C_{2,D} = (12.76\sqrt{2}D)^2(r_0/\delta)^2$.

Corollary C.3. *We note that $B(l)$ from Eq. (5.8) has a maximum at*

$$l^* = \frac{1}{2}\log_2\left(\frac{C_{2,D}\log 4}{D\log 2}\right) = \log_2\left(\sqrt{\frac{D}{2}}\left(\frac{r_0}{\delta}\right)\right) + \log_2\left(\frac{4M_D}{D}\sqrt{2}\right)$$

and is monotonically increasing with l on the interval $l \in (0, l^*)$. Furthermore, with the a fortiori expression for M_D from Remark C.2 we have

$$l^* = \log_2\left(\sqrt{\frac{D}{2}}\left(\frac{r_0}{\delta}\right)\right) + \log_2\left(25.52\sqrt{2}\right).$$

Lemma C.4. *The function*

$$F(T) := 2^{-\frac{T}{2}(1+1/\ln 2)}2^{-lT}T^{\frac{T}{2}}\left(\frac{r_0}{\delta}\right)^T$$

has minimum

$$F(T^*) = 2^{-\frac{1}{\ln 2}4^{l-\log_2(r_0/\delta)}}.$$

Proof. We can write $F(T)$ as

$$F(T) = 2^{-\frac{T}{2}(1+1/\ln 2)}2^{-lT}2^{T/2\log_2 T}2^{T\log_2(r_0/\delta)} = 2^{-T/2(B-\log_2 T)} = 2^{f(T)},$$

where $B = 1 + \frac{1}{\ln 2} + 2l - 2\log_2\left(\frac{r_0}{\delta}\right)$ and $f(T) = -T/2(B - \log_2 T)$. $F(T)$ is therefore minimized when $f(T)$ is minimized. Namely when

$$T^* = 2^{B-1/\ln 2} = 2^{1+1/\ln 2+2l-2\log_2(r_0/\delta)-1/\ln 2} = 2\cdot 4^{l-\log_2(r_0/\delta)}.$$

Inserting this back into the expression for $F(T)$ gives the desired result. □

References

[1] B. Schölkopf, A.J. Smola, Learning with kernels: Support vector machines, regularization, optimization, and beyond, 1st, MIT press, 2002.
[2] A. Rudi, L. Carratino, L. Rosasco, FALKON: An optimal large scale kernel method, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Proc. 31th Int. Conf. Neural Inf. Process. Syst., volume 30, 2017, pp. 3889–3899.
[3] A.E. Alaoui, M.W. Mahoney, Fast randomized kernel ridge regression with statistical guarantees, in: Proc. 28th Int. Conf. Neural Inf. Process. Syst., volume 1, 2015, pp. 775–783.
[4] Y. Zhang, J. Duchi, M. Wainwright, Divide and conquer kernel ridge regression: adistributed algorithm with minimax optimal rates, J. Mach. Learn. Res. 16 (2015) 3299–3340.
[5] H. Avron, K.L. Clarkson, D.P. Woodruff, Faster kernel ridge regression using sketching and preconditioning, J. Matrix. Anal. Appl. 38 (4) (2017) 1116–1138, doi:10.1137/16M1105396.
[6] E. Burnaev, I. Nazarov, Conformalized kernel ridge regression, in: Proc. 15th Int. Conf. Mach. Learn. Appl., 2017, pp. 45–52, doi:10.1109/ICMLA.2016.65.
[7] P. Exterkate, P.J. Groenen, C. Heij, D. van Dijk, Nonlinear forecasting with many predictors using kernel ridge regression, Int. J. Forecas 32 (3) (2016) 736–753, doi:10.1016/j.ijforecast.2015.11.017.
[8] M. Niu, S. Rogers, M. Filippone, D. Husmeier, Fast parameter inference in nonlinear dynamical systems using iterative gradient matching, in: Proc. 33rd Int. Conf. Mach. Learn. Res., 2016, pp. 1699–1707.
[9] M. Stock, T. Pahikkala, A. Airola, B. De Baets, W. Waegeman, A comparative study of pairwise learning methods based on kernel ridge regression, Neural Comput. 30 (8) (2018) 2245–2283, doi:10.1162/neco_a_01096.
[10] S. An, W. Liu, S. Venkatesh, Face recognition using kernel ridge regression, in: Proc. Conf. Comput. Vis. Recognit., 2007, pp. 1–7, doi:10.1109/CVPR.2007.383105.
[11] B.Y.S. Li, L.F. Yeung, K.T. Ko, Indefinite kernel ridge regression and its application on QSAR modelling, Neurocomputing 158 (2015) 127–133, doi:10.1016/j.neucom.2015.01.060.
[12] P. Mohapatra, S. Chakravarty, P.K. Dash, Microarray medical data classification using kernel ridge regression and modified cat swarm optimization based gene selection system, Swarm. Evol. Comput. 28 (2016) 144–160, doi:10.1016/j.swevo.2016.02.002.
[13] S. Muthukrishnan, Data streams: algorithms and applications, Found. Trends Theor. Comput. Sci. 1 (2) (2005) 117–236, doi:10.1561/0400000002.
[14] W. Fan, A. Bifet, Mining big data, ACM SIGKDD Explor. Newsl. 14 (2) (2013) 1–5, doi:10.1145/2481244.2481246.
[15] K. Lan, D.-T. Wang, S. Fong, L.-S. Liu, K.K.L. Wong, N. Dey, A survey of data mining and deep learning in bioinformatics, J. Med. Syst. 42 (8) (2018), doi:10.1007/s10916-018-1003-9.
[16] J. Kivinen, A.J. Smola, R.C. Williamson, Online learning with kernels, in: Proc. 14th Int. Conf. Neural Inf. Process. Syst., 2001, pp. 785–792, doi:10.7551/mitpress/1120.003.0105.

[17] C. Scovel, D. Hush, I. Steinwart, J. Theiler, Radial kernels and their reproducing kernel hilbert spaces, J. Complex. 26 (6) (2010) 641–660, doi:10.1016/j.jco.2010.03.002.
[18] C.A. Micchelli, Y. Xu, H. Zhang, Universal kernels, J. Mach. Learn. Res. 7 (2006) 2651–2667.
[19] Z. Wang, K. Crammer, S. Vucetic, Breaking the curse of kernelization: budgeted stochastic gradient descent for large-scale SVM training, J. Mach. Learn. Res. 13 (2012) 3103–3131.
[20] C.R. Loader, Bandwidth selection: classical or plug-in? Ann. Stat. 27 (2) (1999) 415–438.
[21] G.C. Cawley, N.L. Talbot, Fast exact leave-one-out cross-validation of sparse least-squares support vector machines, Neural Netw. 17 (10) (2004) 1467–1475, doi:10.1016/j.neunet.2004.07.002.
[22] S. Arlot, A. Celisse, A survey of cross-validation procedures for model selection, Stat. Surv. 4 (2010) 40–79, doi:10.1214/09-SS054.
[23] R. Krauthgamer, J.R. Lee, Navigating nets: Simple algorithms for proximity search, in: Proc. 15th Annu. ACM-SIAM Symp. Discrete Algorithms, 2004, pp. 798–807.
[24] A. Beygelzimer, S. Kakade, J. Langford, Cover trees for nearest neighbor, in: Proc. 23th Int. Conf. Mach. Learn., 2006, pp. 97–104, doi:10.1145/1143844.1143857.
[25] T. Shinbrot, C. Grebogi, J. Wisdom, J.A. Yorke, Chaos in a double pendulum, Am. J. Phys. 60 (6) (2016) 491–499, doi:10.1119/1.16860.
[26] A. Marcelo Tusset, V. Piccirillo, A.M. Bueno, J. Manoel Balthazar, D. Sado, J.L.P. Felix, R.M.L. Brasil, Chaos control and sensitivity analysis of a double pendulum arm excited by an RLC circuit based nonlinear shaker, J. Vib. Control 22 (17) (2016) 3621–3637, doi:10.1177/1077546314564782.
[27] N. Rabin, R.R. Coifman, Heterogeneous datasets representation and learning using diffusion maps and Laplacian pyramids, in: Proc. 12th Int. Conf. Data Min., 2012, pp. 189–199, doi:10.1137/1.9781611972825.17.
[28] W. Leeb, Properties of Laplacian pyramids for extension and denoising, 2019, arXiv:1909.07974
[29] P.J. Burt, E.H. Adelson, The laplacian pyramid as a compact image code, IEEE Trans. commun. 31 (4) (1983) 532–540.
[30] G.R. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, M.I. Jordan, Learning the kernel matrix with semidefinite programming, J. Mach. Learn. Res. 5 (2004) 27–72.
[31] F.R. Bach, G.R. Lanckriet, M.I. Jordan, Multiple kernel learning, conic duality, and the SMO algorithm, in: Proc. 21th Int. Conf. Mach. Learn., 2004, pp. 41–48, doi:10.1145/1015330.1015424.
[32] S. Sonnenburg, G. Rätsch, C. Schäfer, B. Schölkopf, Large scale multiple kernel learning, J. Mach. Learn. Res. 7 (2006) 1531–1565.
[33] E.G. Băzăvan, F. Li, C. Sminchisescu, Fourier kernel learning, in: Eur. Conf. Comput. Vis., 2012, pp. 459–473, doi:10.1007/978-3-642-33709-3_33.
[34] A. Bermanis, A. Averbuch, R.R. Coifman, Multiscale data sampling and function extension, Appl. Comput. Harmon. Anal. 34 (1) (2013) 15–29, doi:10.1016/j.acha.2012.03.002.
[35] N. Rabin, D. Fishelov, Multi-scale kernels for nystr̨m based extension schemes, Appl. Math. Comput. 319 (2018) 165–177, doi:10.1016/j.amc.2017.02.025.
[36] W. Liao, M. Maggioni, S. Vigogna, Multiscale regression on unknown manifolds, Mathematics in Engineering 4 (4) (2022) 1–25, doi:10.3934/mine.2022028.
[37] H. Fan, Q. Song, S.B. Shrestha, Kernel online learning with adaptive kernel width, Neurocomputing 175 (2015) 233–242, doi:10.1016/j.neucom.2015.10.055.
[38] B. Chen, J. Liang, N. Zheng, J.C. Príncipe, Kernel least mean square with adaptive kernel size, Neurocomputing 191 (2016) 95–106, doi:10.1016/j.neucom.2016.01.004.
[39] J. Zhang, H. Ning, X. Jing, T. Tian, Online kernel learning with adaptive bandwidth by optimal control approach, IEEE Trans. Neural Netw. Learn. Syst. 32 (5) (2021) 1920–1934, doi:10.1109/TNNLS.2020.2995482.
[40] A. Graps, An introduction to wavelets, IEEE Comput. Sci. Eng. 2 (2) (1995) 50–61, doi:10.1109/99.388960.
[41] A.N. Akansu, W.A. Serdijn, I.W. Selesnick, Emerging applications of wavelets: a review, Phys. Commun. 3 (1) (2010) 1–18, doi:10.1016/j.phycom.2009.07.001.
[42] R.R. Coifman, M. Maggioni, Diffusion wavelets, Appl. Comput. Harmon. Anal. 21 (1) (2006) 53–94, doi:10.1016/j.acha.2006.04.004.
[43] M. Maggioni, H.N. Mhaskar, Diffusion polynomial frames on metric measure spaces, Appl. Comput. Harmon. Anal. 24 (3) (2008) 329–353, doi:10.1016/j.acha.2007.07.001.
[44] D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory, Appl. Comput. Harmon. Anal. 30 (2) (2011) 129–150, doi:10.1016/j.acha.2010.04.005.
[45] A. Cloninger, H. Li, N. Saito, Natural graph wavelet packet dictionaries, J. Fourier Anal. Appl. 27 (3) (2021) 1–33, doi:10.1007/s00041-021-09832-3.
[46] E. De Vito, Z. Kereta, V. Naumova, L. Rosasco, S. Vigogna, Wavelet frames generated by a reproducing kernel, J. Fourier Anal. Appl. 27 (2) (2021) 1–39, doi:10.1007/s00041-021-09835-0.
[47] A. Rahimi, B. Recht, Random features for large-scale kernel machines, in: Adv. Neural Inf. Process Syst., volume 20, 2008, pp. 1177–1184.
[48] Q.V. Le, T. Sarlos, A. Smola, Fastfood-computing hilbert space expansions in loglinear time, in: Proc. 30th Int. Conf. Mach. Learn., volume 28, 2013, pp. 244–252.
[49] Z. Yang, A.J. Smola, L. Song, A.G. Wilson, A la carte | learning fast kernels, in: Proc. 18th Int. Conf. Artif. Intell. Stat., volume 38, 2015, pp. 1098–1106.
[50] J. Zhang, A. Cloninger, R. Saab, Sigma-delta and distributed noise-shaping quantization methods for random fourier features, 2021, arXiv:2106.02614
[51] C. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: Proc. 14th Annu. Conf. Neural Inf. Process Syst., volume 13, 2001, pp. 682–688.
[52] A.J. Smola, B. Schölkopf, Sparse greedy matrix approximation for machine learning, in: Proc. 1th Int. Conf. Mach. Learn., 2000, pp. 911–918.
[53] A. Cloninger, Prediction models for graph-linked data with localized regression, in: Proc. SPIE Int. Soc. Opt. Eng., volume 10394, 2017, doi:10.1117/12.2271840.
[54] S. Ma, R. Bassily, M. Belkin, The power of interpolation : Understanding the effectiveness of SGD, in: Proc. 35th Int. Conf. Mach. Learn., 2018, pp. 3331–3340.
[55] S. Ma, M. Belkin, Kernel machines that adapt to GPUs for effective large batch training, 2018, arXiv:1806.06144
[56] J. Picka, Statistical inference for disordered sphere packings, Stat. Surv. 6 (2012) 74–112, doi:10.1214/09-SS058.
[57] M. Hifi, R. M'Hallah, A literature review on circle and sphere packing problems: models and methodologies, Adv. Oper. Res. (2009), doi:10.1155/2009/150624.
[58] P. Flajolet, D. Gardy, L. Thimonier, Birthday paradox, coupon collectors, caching algorithms and self-organizing search, Discrete Appl. Math. 39 (3) (1992) 207–229, doi:10.1016/0166-218X(92)90177-C.
[59] G. Klambauer, Problems and propositions in analysis, Marcel Dekker, New York, 1979.
[60] H. Wendland, Scattered data approximation, Cambridge University Press, 2004, doi:10.1017/cbo9780511617539.
[61] D. Gómez, A more direct proof of gerschgorinós theorem, Mat: Enseñanza Univ. 14 (2) (2006) 119–122.
[62] T. Hofmann, B. Schölkopf, A.J. Smola, Kernel methods in machine learning, Ann. Stat. 36 (3) (2008) 1171–1220, doi:10.1214/009053607000000677.
[63] M. Aiserman, E.M. Braverman, L.I. Rozonoer, Theoretical foundations of the potential function method in pattern recognition, Avtomat. i Telemeh. 25 (6) (1964) 917–936.
[64] B.E. Boser, I.M. Guyon, V.N. Vapnik, A training algorithm for optimal margin classifiers, in: Proc. 5th Annu. Workshop. Comput. Learn. Theory, 1992, pp. 144–152, doi:10.1145/130385.130401.
[65] G.S. Kimeldorf, G. Wahba, A correspondence between bayesian estimation on stochastic processes and smoothing by splines, Ann. Math. Stat. 41 (2) (2011) 495–502, doi:10.1214/aoms/1177697089.
[66] B. Schölkopf, R. Herbrich, A.J. Smola, A generalized representer theorem, in: Int. Conf. Comput. Learn. Theory, 2001, pp. 416–426, doi:10.1007/3-540-44581-1_27.

Paper II

Improving inversion of model parameters from action potential recordings with kernel methods

Andreas Oslandsbotn, **Alexander Cloninger**
Nickolas Forsch

Submitted to Mathematical Biosciences and Engineering
(BioRxiv: doi.org/10.1101/2023.03.15.532862)

Paper III

Effective resistance in metric spaces

Robi Bhattacharjee, **Alexander Cloninger**
Yoav Freund, **Andreas Oslandsbotn**

Submitted to Journal of Machine Learning Research

(Co-first-authors: Robi Bhattacharjee and Andreas Oslandsbotn)

(arXiv: doi.org/10.48550/arXiv.2306.15649)

Paper IV

Structure from voltage

Robi Bhattacharjee, **Alexander Cloninger**,
Yoav Freund, **Andreas Oslandsbotn**

In preparation for submission

(Co-first-authors: Robi Bhattacharjee and Andreas Oslandsbotn)

(arXiv: doi.org/10.48550/arXiv.2203.00063)