Learning patterns with kernels and learning kernels from patterns

Thesis by
Gene Ryan Yoo

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

Caltech

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2020
Defended September 2, 2020

ACKNOWLEDGEMENTS

I would like to offer my deep thanks to my advisor, Houman Owhadi. The knowledge and wisdom I received while working with him will surely pay dividends in the future. When I started working with him, I was unsure if I could contribute to his research projects. With his guidance and patience, I was able to strengthen my many weaknesses and develop as a researcher. He believed in me when few did. He probably changed the course of my life. I would also like to thank the members of my committee—Andrew Stuart, Nikolai Markovic, Omer Tamuz, and Peter Schroeder—for their very helpful input during our meetings. Additionally, I thank Vladomir Markovic for suggesting my transition from pure to applied math and Dinakar Ramakrishnan for his support and encouragement.

I also owe a debt of gratitude to my family. My mother was always there, cheering me on no matter how difficult things became. I thank her for pushing me to attempt a PhD. Furthermore, I thank my loving maternal grandparents for always cooking for me and keeping my refrigerator filled with food. I also give thanks to my brother for always keeping my journey entertaining.

I am extremely grateful as well to the many teachers who played a large part in shaping my academic career. Ali Gurel, Vjekoslav Kovac, Tonci Crmaric, and Nandlal Jhunjhunwala taught me almost everything I know about working with mathematics problems. Fatih Gelgi gave me the tools I needed to implement nearly any computer code I needed during the course of my PhD research. I appreciate all of my teachers and professors at Seoul Foreign School, Roscomare Elementary School, Mirman School, UCLA, and MIT. Mrs. Jefferd and Mr. Fren first recognized and fostered my love for mathematics. Mr. Wiener always believed in me, guided me while growing up, and cemented my mathematical interest. Mr. Brennan, Mr. Brady, and Mr. Sewell's tough love helped me manage my horrid organizational skills later in my life. Olga Radko made the excellent recommendation that I take courses at UCLA to expand my horizons. She is missed and my deepest condolences to her family. Graciela Gelmini helped me in my transition to UCLA and shaped the way I think about physics. Thank you as well to the great people at the intern program at Susquehanna International Group for the valuable learning experiences and the recommendation to hone my craft in graduate school. I look forward to returning.

Finally, I thank my amazing colleges at Caltech, the friends I made playing basketball and baseball, as well as Phil, Larry, and Kurt for the wisdoms and shenanigans. Finally I'm eternally grateful to my childhood friends Darwin, Alina, and David, for sticking with me over all these years. A special thank you to Alina and Karl Hagen for helping me proofread my thesis.

ABSTRACT

A major technique in learning involves the identification of patterns and their use to make predictions. In this work, we examine the symbiotic relationship between patterns and Gaussian process regression (GPR), which is mathematically equivalent to kernel interpolation. We introduce techniques where GPR can be used to learn patterns in denoising and mode (signal) decomposition [102, 153]. Additionally, we present the kernel flow (KF) algorithm which learns a kernels from patterns in the data [103] with methodology inspired by cross validation. We further show how the KF algorithm can be applied to artificial neural networks (ANNs) to make improvements to learning patterns in images [154].

In our denoising and mode decomposition examples, we show how kernels can be constructed to estimate patterns that may be hidden due to data corruption. In other words, we demonstrate how to learn patterns with kernels. Donoho and Johnstone [38] proposed a near-minimax method for reconstructing an unknown smooth function u from noisy data $u + \zeta$ by translating the empirical wavelet coefficients of $u + \zeta$ towards zero. We consider the situation where the prior information on the unknown function u may not be the regularity of u, but that of $\mathcal{L}u$ where $\mathcal{L}$ is a linear operator, such as a partial differential equation (PDE) or a graph Laplacian. We show that a near-minimax approximation of u can be obtained by truncating the $\mathcal{L}$-gamblet (operator-adapted wavelet) coefficients [101] of $u + \zeta$. The recovery of u can be seen to be precisely a Gaussian conditioning of $u + \zeta$ on measurement functions with length scale dependent on the signal-to-noise ratio.

We next introduce kernel mode decomposition (KMD), which has been designed to learn the modes $v_i = a_i(t)y_i\big(\theta_i(t)\big)$ of a (possibly noisy) signal $\sum_i v_i$ when the amplitudes a_i, instantaneous phases θ_i, and periodic waveforms y_i may all be unknown. GPR with Gabor wavelet-inspired kernels is used to estimate a_i, θ_i, and y_i. We show near machine precision recovery under regularity and separation assumptions on the instantaneous amplitudes a_i and frequencies $\dot{\theta}_i$.

GPR and kernel interpolation require the selection of an appropriate kernel modeling the data. We present the KF algorithm, which is a numerical-approximation approach to this selection. The main principle the method utilizes is that a “good” kernel is able to make accurate predictions with small subsets of a training set. In this way, we learn a kernel from patterns. In image classification, we show that the learned kernels are able to classify accurately using only one training image per class and show signs of unsupervised learning. Furthermore, we introduce the combination of the KF algorithm with conventional neural-network training. This combination is able to train the intermediate-layer outputs of the network simultaneously with the final-layer output. We test the proposed method on Convolutional Neural Networks (CNNs) and Wide Residual Networks (WRNs) without alteration of their structure or their output classifier. We report reduced test errors, decreased generalization gaps, and increased robustness to distribution shift without significant increase in computational complexity relative to standard CNN and WRN training (with Drop Out and Batch Normalization).

As a whole, this work highlights the interplay between kernel techniques with pattern recognition and numerical approximation.

PUBLISHED CONTENT AND CONTRIBUTIONS

[1] H. Owhadi, C. Scovel, and G.R. Yoo, *Kernel Mode Decomposition and programmable/interpretable regression networks*, arXiv preprint arXiv:1907.08592 (2019),
G.R.Y. was involved in developing and implementing the non-trigonometric and unknown waveform Kernel Mode Decomposition Networks and the writing of the corresponding sections in the manuscript. G.R.Y. also developed and implemented the segmented KMD algorithm which has robustness crossing frequency and vanishing modes as well as noise. G.R.Y. contributed to the proving of the results on the universality of the aggregated kernel.

[2] H. Owhadi and G.R. Yoo, *Kernel flows: From learning kernels from data into the abyss*, Journal of Computational Physics **389** (2019), 22–47, https://doi.org/10.1016/j.jcp.2019.03.040
G.R.Y. was involved in developing and implementing the application of Kernel Flows to Convolutional Neural Networks as well as writing the corresponding section in the manuscript.

[3] G. R. Yoo and H. Owhadi, *De-noising by thresholding operator adapted wavelets*, Statistics and Computing **29** (2019), no. 6, 1185–1201, https://doi.org/10.1007/s11222-019-09893-x
G.R.Y. was involved in developing the denoising algorithm, proving the near-minimax recovery theorems, computing implementations, and writing of the manuscript with the help and guidance of H.O.

[4] G.R. Yoo and H. Owhadi, *Deep regularization and direct training of the inner layers of neural networks with kernel flows*, arXiv preprint arXiv:2002.08335 (2020),
G.R.Y. invented and implemented the CNN variant simultaneously training inner layer and network outputs with the use of KF. Further involvement included the writing of the manuscript with help and guidance from H.O.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Chapter 1

INTRODUCTION

The supervised learning problem involves estimating the relationship between a pair of variables based on a finite number of measurements and making predictions with this estimated relationship. This problem can be approached by utilizing known underlying tendencies, laws, or patterns at play. In addition, solving the learning problem can lead to clues to improve such knowledge. These observations relate to main themes of this thesis, which are *learning patterns with kernels* and *learning kernels from patterns*.

Kriging and *Gaussian process regression* (GPR) are flexible tools with strong mathematical theory that have been used to address the supervised learning problem. Section 1.1 will give more details on kriging and GPR. Although they are derived using differing assumptions, they are mathematically identical. GPR, which is also known as *kernel interpolation*, involves conditioning model Gaussian processes to make predictions. The mathematical theory has been well known since at least the 1940s, but was not regularly used in applications until the 1970s [109, Sec. 2.8]. Meanwhile, in the Geostatics field, kriging was developed in the early 1950s (without Gaussian Process theory) by Danie Krige with the goal of calculating unbiased estimates in the mapping of natural resource potential [16, 79].

Kriging is commonly used in geological mapping applications, such as soil analysis [88] and satellite image classification of land [115], as well as in the research of air quality networks such as the justification of optimal sensor locations [10] and the use of low-cost sensors which may have both poorer accuracy than their traditional counterparts and data gaps in time and space [1, 118]. Another application is in the mapping of the distribution of metals in galaxies to gain clues to the process of their formation [12, 106]. All of these mapping applications are regressions based on spatial data.

One-dimensional signal analysis problems can also be approached with GPR, an overview of which can be found in [113]. Examples include detecting the periodicity in weather and climate data [98] as well as the quasi-periodicity in stellar cycles [2]. GPR is also used in detecting changepoints (i.e., points in the signal where the characteristics change suddenly). Examples of this use include detecting changes in disease incidence or stock market data [113]. More uses in the physical sciences can be found in [16].

In addition to these physical applications, GPR can be used in the general classification or regression of data, which is also known as supervised machine learning. One such example is creating automated real-estate price appraisals based on data including square footage, condition, and age [81]. GPR is also applied in sports analytics models. It is used in the estimation of the rate of injuries and recovery time in soccer [71]. It is also used in the regression of probabilities of outcomes in each National Basketball Association game based on individual player statistics [82]. Details on the use of GPR and comparisons to its kernel method relative, Support Vector Machines (SVM), in the context of machine learning can be found in [109, 120]. Within this genre, the canonical examples involve the construction of an image classifier with the MNIST or CIFAR databases. Each is divided into 10 classes, where MNIST consists of images of each of the 10 written numerical digits. CIFAR image classes include 6 various types of animals and 4 types of vehicles. Between the two, GPR methods are much more successful in the MNIST database, with performances approaching those of modern Neural Network (NN) approaches [136, 156].

1.1 Theory of kriging and Gaussian process regression

The supervised learning problem can be formulated as follows:

Problem 1. *Suppose $u : \mathcal{X} \to \mathcal{Y}$ is unknown. Given x_i and $y_i = u(x_i)$ for $i = 1, \ldots, N$, estimate u.*

The spaces $\mathcal{X}$ and $\mathcal{Y}$ are referred to as the input and output spaces respectively. The observations $(x_i, u(x_i))$ are designated the training data set.[1] In many contexts, a testing set $(x_i^t, u(x_i^t))$ is defined to quantify the accuracy of the estimated relation. Explicitly, a loss function dependent on $u^*(x_i^t) - u(x_i^t)$ is used, such as $\sum_i \left(u^*(x_i^t) - u(x_i^t)\right)^2$. In interpolation problems, u^* agrees with u over the training set, i.e., $u^*(x_i) = u(x_i)$. In contrast, regression problems have a slightly altered assumption where observations of the output are noisy, i.e., $(x_i, u(x_i) + \zeta_i)$ for random, independent noise ζ_i.

[1] It is common to refer to $u(x_i)$ as the label of x_i.

Simple kriging

As described in [97], there are multiple variants of kriging. We first discuss the mathematics behind *Simple kriging*. It is assumed that $\mathbf{X} : \Omega \to \mathbb{R}$ is a random real-valued function on some domain Ω (typically $\Omega \subset \mathbb{R}^n$) with known finite mean and covariance functions $m(t) = \mathbb{E}[X_t]$ and $k(t, t') = \mathrm{Cov}(X_t, X_{t'})$. There are no assumptions made on the distribution of $\mathbf{X}$ beyond its mean and covariance. While there are no restrictions on the mean function m, the covariance k must be such that $\mathbf{K}(D, D) := (k(t^i, t^j))_{1 \le i,j \le n}$ is a positive definite $n \times n$ matrix for any finite subset $D := \{t^1, \ldots, t^n\}$ of Ω with distinct elements. Any such function k is called *valid* or *positive definite*. We further assume that we have partial measurements of $\mathbf{X}$ at a finite number of measurement points $\{t^1, \ldots, t^n\} = D \subset \Omega$ (i.e., the *training set*). Note that, for simplicity, when only one training data set is used, we write $\mathbf{K} = \mathbf{K}(D, D)$. For any $t \in \Omega$, the simple kriging classifier is a linear combination of the realized values of $\mathbf{X}$ on D:

$$\hat{X}_t = m(t) + \sum_{t^i \in D} w^i(t)\big(\mathbf{X}_{t^i} - m(t^i)\big)\,, \tag{1.1.1}$$

where the weights $\mathbf{w}(t)$ are selected to minimize the variance of the estimation error, $\mathrm{Var}(X_t - \hat{X}_t)$. This variance can be calculated using covariance function k:

$$\mathrm{Var}(X_t - \hat{X}_t) = k(t, t) - 2\mathbf{k}(t, D)\mathbf{w}(t) + \mathbf{w}(t)^\top \mathbf{K}\mathbf{w}(t)\,, \tag{1.1.2}$$

with $\mathbf{k}(t, D) = (k(t, t^i))_{1 \le i \le n} \in \mathbb{R}^{1 \times n}$ and the last two terms of the left hand side consisting of standard matrix multiplication. To minimize with respect to $\mathbf{w}(t)$, we solve the normal equations and obtain

$$\mathbf{w}(t) = \mathbf{k}(t, D)\mathbf{K}^{-1}\,, \tag{1.1.3}$$

and hence the simple kriging interpolator is given by

$$\hat{X}_t = m(t) + \mathbf{k}(t, D)\mathbf{K}^{-1}\big(\mathbf{X}_D - \mathbf{m}(D)\big)\,, \tag{1.1.4}$$

where $\mathbf{X}_D = \{X_{t^1}, \ldots, X_{t^n}\}$ and $\mathbf{m}(D) = \{m(t^1), \ldots m(t^n)\}$ are both $\mathbf{R}^{n \times 1}$ vectors. The kriging solution to the supervised learning problem is $u^*(t) = \hat{X}_t$. Extensions of this method include accommodating the altered assumptions that the mean is unknown (i.e., ordinary or universal kriging [97, Ch. 4,6]) or that random function X is vector-valued (i.e., cokriging [97, Ch. 13]). Note that both are equivalent to special cases of Gaussian Process Regression, summarized in what follows.

Gaussian process regression

We begin by reviewing the mathematical theory of Gaussian processes and their regressions. It is shown that this regression with Gaussian assumptions yields the identical result as simple kriging. It is also pointed out that Gaussian process regressions in the first form presented in this subsection are actually interpolations. Regressions using GPR can be accomplished with smoothing as presented in 1.1.12. We begin by defining *Gaussian vectors*.

Definition 1.1.1. *A $\mathbb{R}^d$-valued random vector, $\mathbf{X}$, is a normal or Gaussian random vector with mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance $\mathbf{K} \in \mathbb{R}^{d\times d}$, for $\mathbf{K}$ a positive definite matrix, if its probability distribution is given by*

$$P(\mathbf{X} = \mathbf{y}) = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\mathbf{K})}} \exp[-\frac{1}{2}(\mathbf{y} - \boldsymbol{\mu})^\top \mathbf{K}^{-1}(\mathbf{y} - \boldsymbol{\mu})] \,. \tag{1.1.5}$$

Any such $\mathbf{X}$ is denoted by $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ and the mean and covariance of $\mathbf{X}$ is indeed $\mathbb{E}[\mathbf{X}] = \boldsymbol{\mu}$ and $\mathrm{Cov}(\mathbf{X}) = \mathbf{K}$.

Note that there exists a Gaussian vector with arbitrary mean and valid covariance matrix with unique distribution. Moreover, Gaussian vectors have many useful properties. For example, the conditional distributions of any two components is Gaussian. Furthermore, there are more fundamental results such as the Central Limit theorem. This theorem states that for arbitrary identically independently distributed (IID) X_i with finite variance,

$$\frac{\sum_{i=1}^n X_i - \mathbb{E}[X_i]}{\sqrt{n}} \tag{1.1.6}$$

converges to a Gaussian vector. We next define a *Gaussian process* (GP).

Definition 1.1.2. *A collection of random variables, $\{X_t\}_{t\in\Omega}$, indexed by arbitrary set T, is a Gaussian process if for every finite subset with distinct elements of Ω, $\{t^1, t^2, \dots, t^n\}$, the $\mathbb{R}^n$-valued random vector $(X_{t^1}, X_{t^2}, \dots, X_{t^n})$ is a Gaussian random vector.*

We use the identical notation as used in kriging. The mean and covariances of a GP, X, are expressed here by functions $m : T \longrightarrow \mathbb{R}$ and $k : T \times T \longrightarrow \mathbb{R}$, respectively, where $m(t) = \mathbb{E}[X_t]$ and $k(t, t') = \mathrm{Cov}(X_t, X_{t'})$ is a valid covariance. Note that there exists a unique GP, up to distribution, with mean m and valid covariance function k, which is written $X \sim \mathcal{N}(m, k)$. Note that occasionally k is expressed as $T \longrightarrow \mathbb{R}$ function, which implies a stationary kernel $k(t, t') = k(|t - t'|)$.

Shifting to the theory of regression with Gaussian processes, we examine a natural question: given a GP $X \sim \mathcal{N}(m, k)$ with the observation of X over finite subset $D = \{t^1, \ldots, t^n\} \subset T$, i.e., $\mathbf{X_D} := (X_{t^1}, \ldots, X_{t^n})^\top$, what can we say about the distribution of X_t for arbitrary $t \in T$? It is well known that the conditional distributions of Gaussian vectors are also Gaussian, with distributions that are computable using the fact that L^2 orthogonality is equivalent to independence [101, Thm. 7.2]. More explicitly,

$$X_t|\mathbf{X_D} = \left(X_t - \mathbf{k}(t, D)\mathbf{K}^{-1}\mathbf{X_D}\right) + \mathbf{k}(t, D)\mathbf{K}^{-1}\mathbf{X_D}\Big|\mathbf{X_D}\,, \tag{1.1.7}$$

where $\mathbf{k}(t, D) = (k(t, t^i))_{1 \le i \le n} \in \mathbb{R}^{1\times n}$ and $\mathbf{K} = (k(t^i, t^j))_{1 \le i,j \le n} \in \mathbb{R}^{n\times n}$. It can be shown that $X_t - \mathbf{k}(t, D)\mathbf{K}^{-1}\mathbf{X_D}$ is independent to each X_{t^i} for all $t^i \in D$, implying that $X_t|\mathbf{X_D} = \mathbf{y}$ has the same distribution as $X_t - \mathbf{k}(t, D)\mathbf{K}^{-1}(\mathbf{X_D} + \mathbf{y})$. Further calculations can determine its mean and variance:

$$\mathbb{E}[X_t|\mathbf{X_D} = \mathbf{y}] = m(t) + \mathbf{k}(t, D)\mathbf{K}^{-1}(\mathbf{y} - \mathbf{m}(D)) \tag{1.1.8}$$

and

$$\mathrm{Var}(X_t|\mathbf{X_D} = \mathbf{y}) = k(t, t) - \mathbf{k}(t, D)\mathbf{K}^{-1}\mathbf{k}(D, t)\,, \tag{1.1.9}$$

with $\mathbf{m}(D) = (m(t^i))^\top_{1\le i\le n} \in \mathbb{R}^{n\times 1}$. Note that the conditional expectation in equation (1.1.8) is an unbiased estimator for X_t assuming measurement $\mathbf{X_D} = \mathbf{y}$ and has the same mathematical form as the simple kriging estimator. Note that this estimator is defined for all $t \in T$ and is affine relative to observation $\mathbf{y}$. With the assumption X is a centered, i.e., zero mean, Gaussian process our estimator becomes linear and is simplified to $u^*(t) =: \mathbb{E}[X_t|\mathbf{X_D} = \mathbf{y}] = \mathbf{k}(t, D)\mathbf{K}^{-1}\mathbf{y}$.

Gaussian process regression also can refer to the slightly more general context where it is assumed that model GP $X \sim \mathcal{N}(m(t), k(t, t')+\sigma^2\delta_{t,t'})$ is the sum of independent GPs $X^s \sim \mathcal{N}(m(t), k(t, t'))$ and $X^\sigma \sim \mathcal{N}(0, \sigma^2\delta_{t,t'})$, where $\delta_{t,t'} = 1$ when $t = t'$ and $\delta_{t,t'} = 0$ otherwise. Inspired from the fact that virtually all measurements contain error, X is constructed as the sum of a GP modeling the true signal, X^s, and a GP modeling noise, X^σ. This model is an example of additive Gaussian processes, which will be discussed further in Sec. 1.4. The conditional expectation of X^s_t based on measurements of $X = X^s + X^\sigma$ is referred to as a Gaussian process regression (e. g., in [109, Ch. 2]). Further, the estimator given in equation (1.1.8) is a special case of this model with $\sigma = 0$. This conditional expectation and variance can be computed as

$$\mathbb{E}[X^s_t|\mathbf{X^D} = \mathbf{y}] = m(t) + \mathbf{k}(t, D)(\mathbf{K} + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{m}(D)) \tag{1.1.10}$$

and

$$\mathrm{Var}\left(X_t^s|\mathbf{X}^{\mathbf{D}}=\mathbf{y}\right)=k(t,t)-\mathbf{k}(t,D)(\mathbf{K}+\sigma^2\mathbf{I})^{-1}\mathbf{k}(D,t)\,, \tag{1.1.11}$$

where again, $\mathbf{X}_{\mathbf{D}}=(X_{t^1},\ldots,X_{t^n})$ and $\mathbf{K}=k(D,D)$. This can be shown using the independence of X^s and X^σ, implying $\mathrm{Cov}(X_t^s,\mathbf{X}_{\mathbf{D}})=k(t,D)$, and then using the same logic as the original GPR derivation. A full derivation of this result can be found in [109, pg. 16-17]. This form of GPR is no longer an interpolation for $\sigma>0$. Most practical applications use a centered GP X^s, i.e., $m(t)=0$, simplifying the regression formula to

$$\mathbb{E}[X_t^s|\mathbf{X}^{\mathbf{D}}=\mathbf{y}]=\mathbf{k}(t,D)(\mathbf{K}+\sigma^2\mathbf{I})^{-1}\mathbf{y}\,. \tag{1.1.12}$$

1.2 Mathematical applications and interplay of GPR

Reproducing kernel Hilbert spaces

We discuss interplays of GPR with other mathematical topics, beginning with its relation to non-stochastic *Reproducing Kernel Hilbert Spaces* (RKHS) and optimal recovery. Suppose $\mathcal{X}$ is an arbitrary set and $\mathcal{H}$ is a Hilbert space[2] comprised of functions $f:\mathcal{X}\to\mathbb{R}$ such that the evaluation functional $\delta_t(f):=f(t)$ is continuous for each $t\in\mathcal{X}$. This continuity condition can be shown to be equivalent to the existence of a kernel function $k:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ such that $\langle f,k(\cdot,t)\rangle=\delta_t(f)=f(t)$. This then leads to kernel $\langle k(\cdot,t),k(\cdot,t')\rangle=k(t,t')$. Furthermore, there is a one-to-one correspondence between positive definite kernels k and RKHS's over set $\mathcal{X}$. Hence, each kernel corresponds to a unique RKHS norm. Further details on the RKHS can be found in [130, Sec. 4].

With inspiration from optimal recovery [92], we can ask the question, with the observations of $\mathbf{f}(D)=(f(t^i))_{1\le i\le n}^\top\in\mathbb{R}^{n\times 1}$ (and $D=\{t^1,\ldots,t^n\}\subset\mathcal{X}$), what is the minimax optimal recovery of f:

$$g^\dagger=\mathrm{argmin}_g\max_f\frac{\|f-g\|^2}{\|f\|^2}\,, \tag{1.2.1}$$

where the norm $\|\cdot\|$ is the RKHS norm, the max is taken over all $f\in\mathcal{H}$ and the min is taken over $g\in\mathcal{H}$ such that $g(t^i)=f(t^i)$? It is shown in [101, Ch. 8, 18] that the mixed strategy of selecting GP $f^\dagger\sim\mathcal{N}(0,K)$ is a saddle point in the recovery objective function with

$$g^\dagger(t)=\mathbf{k}(t,D)\mathbf{K}^{-1}\mathbf{f}(D)\,, \tag{1.2.2}$$

[2] A complete inner product space.

where we use similar notation setting $k(t, D) = (k(t, t^i))_{1 \leq i \leq n} \in \mathbb{R}^{1 \times n}$ and $K = (k(t^i, t^j))_{1 \leq i,j \leq n}$. Observe this minimax optimal recovery in the RKHS induced norm is precisely GPR on data $(t^i, f(t^i))$.

Support Vector Machines (SVMs) can be utilized in conjunction with an RKHS to construct non-linear regressions or classifiers. This method is commonly referred to as the *kernel method* or *trick* [130]. One similarity with GPR is that the regressions are both linear combinations of $k(t, t^i)$ [130, Thm. 5.5]. This method relies on the fact that for each valid covariance kernel, there exists some feature map $\varphi : \mathcal{X} \to \mathcal{V}$, which may be computationally intractable, into some space $\mathcal{V}$ such that $k(t, t') = \langle \varphi(t), \varphi(t') \rangle_{\mathcal{V}}$. SVM classifiers of t are defined with an affine decision boundary[3] which is determined through optimization. A key point is that the learning of the boundary does not require computations of φ, only of k, which is the motivation for calling this the *kernel trick*.

Links to game theory and optimal recovery

Revisiting the minimax optimality of (1.2.1), a link can be made to game theory. This link can be interpreted as the solution to a two-player, optimal-recovery game [101, Ch. 8]. Supposing Player I has a loss that is the objective function in (1.2.1), while player II has the negative of player I's loss function. Player I selects $f \in \mathcal{H}$ with the aim of maximizing the final recovery error while player II is able to observe $(t^i, f(t^i))$ and selects the g which minimizes recovery error. The optimal strategy of this game is for Player I to implement a mixed strategy of selecting f at random according to $f \sim \mathcal{N}(0, k)$ and for Player II to select g as the GPR recovery with data $(t^i, f(t^i))$ [101, Ch. 8, 18].

Such connections between GPR, game theory, and optimal recovery are made explicit in the exposition of *operator-adapted wavelets* [101] in Theorems 2.2.3 and 2.2.4. These wavelets are interpretable as a Gaussian conditioning on Sobolev spaces (or more generally, Banach spaces) with covariance operator Q [101, Def 7.22, 17.10] and observations corresponding to measurements with functions from the dual space. These measurement functions can be structured in a hierarchy, which leads to a hierarchy of wavelets. These operator-adapted wavelets can also be shown to be an optimal recovery in the norm defined by Q as in [101, Thm. 3.1, 12.4]. Moreover, analogous optimal recovery games in this Banach space setting are also explored in [101, Ch. 8, 18]. These operator-adapted wavelets can be applied to denoise

[3] This boundary lies in space $\mathcal{V}$ and separates points $\varphi(t)$.

solutions to linear equations [153], such as partial differential operators and graph Laplacians, as well as in linear-algebra computations, such as efficient eigenvalue and eigenvector calculation [145] and Cholesky factorization [119].

Smoothing and numerical approximation

We next point out the connection between GPR to interpolation and smoothing by splines. The cubic spline interpolation is a widely used example, where an underlying relationship is estimated by the unique twice-differentiable, piece-wise cubic function with hinge-points at the observed datapoints (t^i, y^i). This can be calculated either analytically or by the minimization of

$$\int (f''(t))^2 dt\,, \tag{1.2.3}$$

with f constrained to agree with observed datapoints. This is shown to be mathematically equivalent to universal kriging in [41]. This can be extended to the minimization of

$$\int (\mathcal{L}f(t))^2 dt\,, \tag{1.2.4}$$

where $\mathcal{L}$ is a linear differential operator, the minimizer of which satisfies $\mathcal{L}^*\mathcal{L}f = 0$ at all points that are not datapoints. Furthermore, when data is known to contain noise, smoothing (regression) splines can be constructed by minimizing

$$\int (\mathcal{L}f(t))^2 dt + \lambda(\mathbf{f}(D) - \mathbf{y})^\top B^{-1}(\mathbf{f}(D) - \mathbf{y}) \tag{1.2.5}$$

for some positive definite B. These splines exchange smoothness and fidelity to the observations according to smoothing parameter λ. It is demonstrated that such splines are mathematically equivalent to GPRs[4] in [75].

Gaussian process regression with a noisy kernel, as in (1.1.12), is used in data smoothing and denoising. This technique is used directly in [157] and [113, Sec. 5d] to study noisy one-dimensional signals. GPR has also been applied to image denoising by using a noisy covariance kernel that is dependent both on pixel position and neighboring pixel intensities in [29, 86]. Section 2 will introduce a novel approach to denoising with GP tools which can be applied to signals, u, where the prior information of the signal may not be the regularity of u but that of $\mathcal{L}u$ for some linear operator $\mathcal{L}$ (such as a PDE or graph Laplacian).

Gaussian process regression, along with more general Bayesian models, can be used in numerical approximation tasks. These applications include solving, optimization,

[4] The model GP has covariance kernel dependent on B and $\mathcal{L}$ as in [75, eq. 3.5].

and quadrature (i.e., numerical integration) [20, 30, 96, 132]. As in the premise of *Information Based Complexity*, the fact that continuous functions cannot be explicitly stored in finite space leads to the need for algorithms which can manipulate them with finite measurements. This partial information interpretation leads to the approach of modeling this known deterministic function with a stochastic method such as GPR.

1.3 GPR in the context of applications

We now discuss the theory supporting GPR in this section in the context of the supervised-learning problem. Moreover, common difficulties in its utilization, namely the specification of the model GP and computational complexity, will be presented in addition to a sample of techniques to address them.

Classification

For virtually all practical examples, an application of GPR involves using equation (1.1.8) to generate an estimator by treating observations $\{(t^i, y^i)\}_{1\leq i\leq n}$ as the realizations of a GP, i.e., $X_{t^i} = y^i$. For simplicity, we make the common assumption that the GP is centered, i.e., that $X \sim \mathcal{N}(0, k)$. In the case of classification with n different classes, the conventional method, inspired from *one-hot encoding*, is to consider n different IID-centered GPs, $X^1, \dots, X^n \sim \mathcal{N}(0, k)$. Then, if data point t^i is identified to be in class c_i, we condition our model GP with[5] $X^k_{t^i} = \delta_{k,c_i}$. Note that taking $X^k_{t^i} = \delta_{k,c_i} - \frac{1}{n}$ can also be used to remove the mean from the outputs. The classifier for arbitrary point t selects the class given by $\operatorname{argmax}_k \mathbb{E}[X^k_t | X^k_{t^i} = \delta_{k,c_i}]$. In the remainder of this chapter, we will revert to the usage of a single model GP for mathematical simplicity, although everything that follows is easily convertible to this context.

Model selection

The first consideration for any application is the choice of model GP. In most situations, a centered GP is assumed (i.e., $m(t) = 0$), in which case only the choice of covariance kernel function must be made. However, for continuous input domains such as $\Omega = \mathbb{R}^d$, this choice has an uncountably infinite degrees of freedom. Intuitively, the aim is to use a kernel $k(t, t')$ which is large precisely when the measurements $X_t = y_t$ and $X_{t'} = y_{t'}$ are expected to be correlated. The contexts of some problems may make this selections relatively clear. For example, in physical

[5] $\delta_{i,j}$ is the Kronecker delta, taking value 1 if $i = j$ and 0 otherwise.

mapping applications, measurements at nearby points may be expected to be better correlated than faraway points. In other applications such as high dimensional image data, it is less clear and we have to learn a kernel. Note that kernels, which are poor metrics of similarity, lead to poor estimators that do not generalize the training set.

Typical approaches for model selection involves starting with a family of kernels, defined as a set of covariance functions $\{k_{\boldsymbol{\theta}}\}$, parameterized by $\boldsymbol{\theta}$, living in some parameter space (typically a $\mathbb{R}^n$ subset). Each kernel $k_{\boldsymbol{\theta}}$ corresponds to a GP $X^{\boldsymbol{\theta}} \sim \mathcal{N}(0, k_{\boldsymbol{\theta}})$. Then an "optimal" kernel in some predefined sense is selected from this family. Note, however, that this approach still requires the choice of kernel family, which is non-obvious in many contexts, such as image recognition or signal analysis. Common choices of kernel families can be found in chapter 4.2 of [109], including the *squared exponential*[6] kernel

$$k(t, t') = \exp\Big(-\frac{|t - t'|^2}{2\sigma^2}\Big), \tag{1.3.1}$$

with a single free parameter σ, which is the length-scale of the GP.

In the context of kriging, the vast majority of geostatistical mapping applications use variogram fitting [17]. Variogram kernels are defined as the function $f(|t-t'|)$ where $k(t, t') = f(|t-t'|)$. A family of variograms are selected with parameters $f(0) > 0$, $\lim_{t\to\infty} f(t)$, and the rate f converges to this limit value. These parameters are known as the nugget, sill, and range of the variogram. Traditionally, these are obtained by dividing all pairwise distances into bins, estimating the variance of pairs in each bin, then parameter fitting.

Maximal likelihood estimates

A common technique for selecting kernel parameters is *Maximal Likelihood Estimation* (MLE) [109, sec. 5.4.1]. As the name suggests, we calculate the likelihood of each kernel in the family $k_{\boldsymbol{\theta}}$, i.e., $P(\mathbf{X}^{\boldsymbol{\theta}}_{\mathbf{D}} = \mathbf{y})$, and maximize $\boldsymbol{\theta}$ over its parameter space. Applying (1.1.5), for $\mathbf{K}_{\boldsymbol{\theta}} = (k_{\boldsymbol{\theta}}(t^i, t^j))_{1\le i,j\le n}$, it holds true that

$$P(\mathbf{X}^{\boldsymbol{\theta}}_{\mathbf{D}} = \mathbf{y}) = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\mathbf{K}_{\boldsymbol{\theta}})}} \exp[-\frac{1}{2}\mathbf{y}^\top {\mathbf{K}_{\boldsymbol{\theta}}}^{-1}\mathbf{y}]\,. \tag{1.3.2}$$

Moreover, maximizing above equation (1.3.2) is equivalent to minimizing

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbf{y}^\top {\mathbf{K}_{\boldsymbol{\theta}}}^{-1}\mathbf{y} + \log(\det(\mathbf{K}_{\boldsymbol{\theta}}))\,, \tag{1.3.3}$$

[6] Also commonly known as the Gaussian or radial basis function (RBF).

which is a linear mapping of the negative log-likelihood of (1.3.2). The optimization of $\mathcal{L}$ is typically non-convex, and parameter spaces with many degrees of freedom usually need an application of gradient descent. In lower-dimensional parameter spaces, the analytical calculation of gradients with Frechet derivatives are usually tractable. Note that although kriging is a distribution-free technique and likelihoods are undefined, MLE is still commonly applied. Due to the mathematical equivalency to GPR, likelihoods of the equivalent GP can be used for kriging covariance kernel learning [53, 105].

The MLE method can be generalized theoretically by examining the parameters in a Bayesian approach. We suppose X is a mixture of random variables, i.e., with probability $P(\theta)$, X has distribution given by $X^{\boldsymbol{\theta}}$. Then with measurements $\mathbf{X_D} = \mathbf{y}$, we can apply Bayesian inference to obtain conditional probabilities

$$P(\boldsymbol{\theta}|\mathbf{y}) = \frac{P(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y})}\,. \tag{1.3.4}$$

Notice that if $p(\boldsymbol{\theta})$ is uniform, then finding the maximal likelihood on $P(\boldsymbol{\theta}|\mathbf{y})$ is equivalent to that of $P(\mathbf{y}|\boldsymbol{\theta})$ as in the MLE approach. Additionally, this leads to model selection that is regularized via the prior, $P(\boldsymbol{\theta})$, where high probabilities are assigned to more "regular" kernels. Other than for the simplest examples, calculating such distributions and their expectations is intractable. Methods such as Monte Carlo, however, can be used to make an estimation. Further details on Bayesian and MLE model selection can be found in [109, 139].

Cross-validation

The generalizeability of a model is a desired quality. This loosely means that the model is not overfitting training data and is able to make predictions. A common method for quantifying this generalizeability is *cross-validation* (CV) [109, Sec. 5.3]. A subset of the training data is designated the validation set and the remaining training data is then used to construct a GPR estimator. The estimation error on the validation set serves as a loss function for CV. While the exposition in this subsection is restricted to the GPR setting, it can be extended to other supervised learning techniques.

A common approach is to use cross-validation on multiple different validation sets, which requires multiple GPR calculations. Typical approaches are k-fold cross-validation, where the training set is split into k disjoint and equal sized subsets. Then every combination of $k - 1$ subsets is used to generate a classifier to estimate validation accuracy on the final subset. Typically k is taken to be approximately 3 to 10. This is implemented by minimizing the following loss with respect to the kernel parameters

$$\mathcal{L}_{\text{CV}}(\boldsymbol{\theta}) = \sum_{i=1}^{k} \log p\left(\mathbf{X}_i^{\boldsymbol{\theta}} = \mathbf{y}_i | \mathbf{X}_{-\mathbf{i}}^{\boldsymbol{\theta}} = \mathbf{y}_{-i}\right), \tag{1.3.5}$$

where $\mathbf{X}_i^{\boldsymbol{\theta}}$ and $\mathbf{X}_{-i}^{\boldsymbol{\theta}}$ refers to the measurements of GP $\mathbf{X}^{\boldsymbol{\theta}}$ on the i-th disjoint subset and all other data points in the training set, respectively. Furthermore, $\mathbf{y}_k$ and $\mathbf{y}_{-k}$ denote the corresponding training subset outputs. This is calculable by estimating the conditional distribution $\mathbf{X}_k^{\boldsymbol{\theta}} | \mathbf{X}_{-k}^{\boldsymbol{\theta}} = \mathbf{y}_{-k}$ as in equation (1.1.7). A notable special case is leave-one-out cross-validation (LOOCV), which takes $k = n$, i.e., the size of the training set. A total of n classifiers are generated from each of the possible selections of $n - 1$ training points to classify the final point. While LOOCV requires n kernel inversions, the fact that each of the interpolation sets differs by a single training point can be exploited in a computational shortcut [109, Eq. 5.12].

The *Kernel Flow* (KF) algorithm [154], presented in Section 6, is a kernel-learning method that is a variant of CV. It operates on the same principle that a kernel is desirable if it is able to accurately generalize a subset of training data to obtain accurate estimates of the labels of the remainder. The technique optimizes kernel parameters with an objective function emulating this interpolation accuracy over various randomly selected training subsets. We obtain data-efficient kernels which are able to compute accurate interpolations with small amounts of the data.

Comparisons between MLE and CV

Some early work comparing of MLE and CV in learning an appropriate spline smoothing factor (i.e., λ in (1.2.5)) can be found in [77, 128, 138]. In the GPR setting, a classical result is that the interpolation with MLE and CV will be asymptotically[7] equivalent within a fixed domain when using a mis-specified covariance kernel[8] [126, 127, 129]. More recent works have been focused on theoretically bounding the approximation error of MLE or CV learned kernels [3–6, 72] with Gaussian or Matérn kernels. These results examine the cases where the true covariance function either lies and does not lie in a specified kernel family. Bounds are obtained in both the fixed-domain and increasing-domain asymptotics on the number of

[7] Asymptotic in number of observation points.

[8] The mis-specified kernel is assumed to be mutually absolutely continuous with the true covariance kernel.

training points. The fixed-domain case assumes that the domain containing the observations is fixed while the latter assumes that the size of the domain increases with training point densities bounded below. The experimental design (i.e., the placement of the training points) is usually taken to be uniform random within the domain or selected with structure, such as Latin hypercube sampling or a grid. On a high level, [3] shows that when the covariance kernel families are correctly specified, the GPR with kernel learned from MLE outperforms the one learned from CV. In contrast, for mis-specified kernel families, CV yields better numerical results than MLE. Most recently, fixed-domain approximation results for kernel parameter learning over compact parameter spaces, which include MLE and CV, without the need of asymptotics have been established in [133, 143]. Note that in typical machine learning applications with high dimensional data, the curse of dimensionality requires the number of training points to be intractably large to observe this asymptotic behavior. Further, the distribution of data points is almost never uniform.

Computational costs

Another important consideration is that the computation of GPR involves the inversion of the kernel matrix, $K = (k(t^i, t^j))_{1\leq i,j\leq n} \in \mathbb{R}^{n\times n}$, where n is the number of training points in our regression. The computational complexity of this matrix inversion is $O(n^3)$ and requires $O(n^2)$ of memory, which quickly becomes prohibitively expensive in large data sets. Furthermore, when applying MLE or CV, there may be a need to evaluate the GPR and possibly its derivative for multiple parameters. This has inspired work on techniques to avoid this explosion of computational costs, an overview of which can be found in chapter 8 of [109] as well as in [47, 85]. The simplest computational shortcut is regressing on a subset of the original training set where the subset is selected either randomly or via some algorithm such as clustering. Others include using sparse kernels, i.e., $k(t, t') = f(|t-t'|)$ which vanishes for sufficiently large $|t-t'|$; obtaining low-rank approximations of K; creating localized models, or “experts,” of the GP estimator; or using an exponential kernel, which has a tri-diagonal inverse [46]. GPR over scalar inputs also can be computationally accelerated to (n) time using Gauss-Markov processes [52, Sec. 1.2] or using *Structured Kernel Interpolation* (SKI) [140], which uses a relatively low number of inducing points to estimate the kernel. Note that SKI is also applicable when the input space is of low dimension, $d \lesssim 5$. Kernels that are *additive* can use these scalar input GPR tricks to obtain (n) complexities and will be discussed further in

Section 1.4.

1.4 Additive Gaussian processes and generalized additive models

This section will review additive Gaussian processes and their connection with the family of generalized additive models. An example of additive GPs has been introduced in the GPR in equation (1.1.12) with a noisy covariance kernel constructed as the sum, or *mixture*, of $X = X^s + X^\sigma$. The conditional distribution of X_t^s with respect to measurements of X are computable as in equations (1.1.10) and (1.1.11).

Gaussian process conditioning of mixtures

With no changes in argument, we can generalize to mixtures of arbitrary independently distributed GP's, i.e., $X^i \in \mathcal{N}(0, k_i)$ with $X = X^1 + \cdots + X^m$. It holds true that

$$\begin{aligned} \mathbb{E}[X_t^i|\mathbf{X_D} = \mathbf{y}] = \mathbf{k_i}(t,D)\Big(\sum_i K_i\Big)^{-1}\mathbf{y} \text{ and} \\ \mathrm{Var}\left(X_t^i|\mathbf{X_D} = \mathbf{y}\right) = k_i(t,t) - \mathbf{k_i}(t,D)\Big(\sum_i K_i\Big)^{-1}\mathbf{k_i}(D,t)\,, \end{aligned} \tag{1.4.1}$$

due to the independence of X^i and $X - X^i$.

Some applications for this framework will be discussed. An example of these additive GPs is *Multiple Kernel Learning* (MKL), which aims to learn the mixture of pre-specified kernels, k_i. These kernels are of the form $k = \sum_i \beta_i k_i$ (with $\beta_i \in \mathbb{R}$) that best models the problem at hand [54]. The separation of periodic and non-periodic components in a signal can be accomplished by applying (1.4.1) to periodic and non-periodic kernels [44]. Many machine learning networks output histogram data, and some classifier of such data is needed. The intersection kernel, which is the sum of pairwise minimum bin counts

$$K_{\text{int}}(x, x') = \sum_{i=1}^{N} \min(x_i, x_i') \tag{1.4.2}$$

applied to a GPR classifier is examined in [114]. This kernel [87], as well as other types of histogram kernels [137], can also be used in an SVM classifier. These kernels are *additive*, defined loosely as each sub-kernel in the mixture being dependent only on one component. They lead to additive models, which will be addressed in the next section.

Additive models

We next examine these additive mixtures of GPs which lead to *additive regressions*, i.e., regressions which can be decomposed into regressions over each component. We suppose that the data input space is $\Omega \subset \mathbb{R}^d$ and that $X \sim \mathcal{N}(0, k)$ with kernel

$$k(t, t') = \sum_{j=1}^{k} k_j(t_j, t'_j)\,, \tag{1.4.3}$$

where t_j represents the j-th component of $t \in \mathbb{R}^d$, meaning that the kernel k_j in the mixture is dependent only on the j-th component of the input data. GPR on $X \sim \mathcal{N}(0, k)$ with an additive kernel leads to the following estimator:

$$\mathbb{E}[X_t | \mathbf{X_D} = \mathbf{y}] = \sum_{j=1}^{d} \mathbf{k_j}(t_j, D)\mathbf{K}^{-1}\mathbf{y}\,, \tag{1.4.4}$$

where $\mathbf{k_j}(t_j, D) = (k_j(t_j, t^i_j))_{1 \le i \le n}$ and $\mathbf{K} := \mathbf{k}(D, D) := (\sum_j k_j(t^i_j, t^k_j))_{1 \le i,k \le n}$. In each term of the sum, the dependence on t is only on a single component[9], and (1.4.4) can be written as

$$\mathbb{E}[X_t | \mathbf{X_D} = \mathbf{y}] = \sum_{j=1}^{d} f_j(t_j)\,. \tag{1.4.5}$$

Such a regression is an example of a *generalized additive model* (GAM), which is defined to be of the form

$$g(f(t)) = \beta + \sum_{j=1}^{d} f_j(t_j)\,, \tag{1.4.6}$$

where link function, g, is smooth monotonic, and hence invertible. It is sometimes taken to be the identity, in which case the regression is also an *additive model*. Another common selection is the logistic function to model probabilities between 0 and 1. Models of this form are favorable due to their easy interpretability, since the contribution to the output from each component of the data space is easily discernable in f_j. Furthermore, data with such additive dependence are less susceptible to the curse of dimensionality. The additive structure can be used to relate datapoints far away in $\mathbb{R}^d$ but with similar j-th components. In other words, a well-sampling of high dimensional spaces no longer requires an exponentially large quantity of data, and kernels of the form $K(t, t') = f(|t - t'|)$ have exponentially

[9] Though there is dependence on other components of training data points, t^i.

fewer pairs of highly correlated points[10] than additive kernels as dimension increases. Additive models also have theoretical motivation due to the Kolmogorov-Arnold representation theorem, which states that every continuous function $f(t_1, \dots, t_d)$ admits an additive form

$$f(t_1, \dots, t_d) = \sum_{q=0}^{2d} \Phi_q\Big(\sum_{p=1}^{d} \phi_{q,p}(t_p) \Big) . \tag{1.4.7}$$

A constructive proof of which can be found in [13]. Further, [124] shows any continuous f is expressible in a restricted additive form

$$f(t_1, \dots, f_d) = \sum_{q=0}^{2d} \Phi\Big(\sum_{p=1}^{d} \lambda_p \phi(t_p + \eta q) + q \Big) , \tag{1.4.8}$$

for a continuous Φ, Lipschitz continuous[11] ϕ, and $\eta, \lambda_p \in \mathbb{R}$.

GAMs have been in the literature for decades. Most classical fitting techniques rely on some variant of backfitting [59, 131]. This method involves iteratively estimating f_j based on $g(f(t)) - \beta - \sum_{i \neq j} f_i(t_i)$, which is interpreted as the residual j-th component of the GAM. Typically, f_i is estimated based on a spline fitting of the residual. More recently, this has been refined to a block-coordinate descent refinement algorithm, in which parameters specifying the GAM are minimized with respect to a loss function iteratively by block [18]. Another recent technique uses *boosting*, which creates multiple GAMs using multiple subsets of the training set. The array of models is then combined, either through averaging or through ensemble voting, leading to a model superior to each of its constituents.

The connection to additive kernel GPR can be seen in methods to avoid the full (n^3) GPR computational cost. An algorithm inspired by backfitting can efficiently compute GPR with an additive kernel, while also using a trick to compute scalar input GPR in (n) complexity [52]. Furthermore, *Structured Kernel Interpolation* (SKI) can be used to approximate scalar-valued kernel k_j, leading to computationally efficient machine-precision estimates of $\mathbf{k_j}(D, D)\mathbf{v}$ [26]. Combining this with the conjugate gradient method leads to GPR that is computable in linear time. Moreover, SKI can be applied to low dimensional input spaces, so it is possible to apply additive GPR to a mixture of sub-kernels that are dependent on approximately $d \lesssim 5$ dimensions of the input space.

[10] Assuming that only nearby datapoints are correlated in the model, i.e., $f(d)$ is only large when d is small.

[11] I.e., continuously differentiable with bounded derivative.

In [43], the comparison is made between additive kernel $K_A = \sum k(t_i, t_i')$ and tensor kernel[12] $K_S = \prod k(t_i, t_i')$, where k is a one-dimensional RBF kernel. It is done by considering the ratio at which the training data reduces variance in testing data estimates, which the Durrande, Ginsbourger, and Roustant term *predictivity*. This predictivity is mathematically defined as

$$P_{k,D} = 1 - \frac{\sum_{t \in D_{\text{test}}} \text{Var}(X_t | \mathbf{X_D})}{\sum_{t \in D_{\text{test}}} \text{Var}(X_t)} = \frac{\sum_{t \in D_{\text{test}}} \mathbf{k}(t, D) \mathbf{K}^{-1} \mathbf{k}(D, t)}{\sum_{t \in D_{\text{test}}} k(t, t)} \,, \tag{1.4.9}$$

for $X \sim \mathcal{N}(0, k)$, with D and D_{test} being finite training and testing sets, respectively. Note that $P_{k,D}$ varies between 0 and 1, where $P_{k,D} = 0$ implies no improvement in the model is made with the knowledge of the training data. On the other hand, $P_{k,D} = 1$ means that testing points are known without uncertainty with training measurements. By comparing $P_{k_A,D}$ and $P_{k_S.D}$ for data sets of differing dimensions, the authors conclude that the additive model is more predictive for higher dimensional sets and the converse for lower numbers of dimensions. They draw similar conclusions by looking at the mixture $k = k_A + k_S$ (with $X \sim \mathcal{N}(0, k)$, $X^A \sim \mathcal{N}(0, k_A)$, and $X^S \sim \mathcal{N}(0, k_S)$), and analogously comparing the ratios of the variances of $X_t^A | \mathbf{X_D}$ with X_t^A and $X_t^S | \mathbf{X_D}$ with X_t^S.

This is generalized in [45], which compares m-th order interaction terms,

$$k_m(t, t') = \sum_{1 \le j_1 \le \cdots \le j_m \le n} \prod_{j_l} k(t_{j_l}, t'_{j_l}) \,, \tag{1.4.10}$$

with full kernel given by the mixture of these terms,

$$k(t, t') = \sum_{m=1}^{d} \sigma_m^2 k_m(t, t') \,. \tag{1.4.11}$$

MLE is used to estimate σ_m, the results of which are interpretable as information on which order terms are the most relevant in modeling the data. Notice that the first and d-th order terms are additive and tensor kernels, as in the previous example. This work shows that the first-order additive kernel does not model the structure of the data for certain real-world examples as effectively as higher order kernels.

This additivity can be generalized from being along the t_i-axes to general directions in $T = \mathbb{R}^d$. A greedy-type algorithm called *projection pursuit GPR* (PPGPR) to iteratively learn projection directions and parameters in kernel

$$k(t, t') = \sum_{j=1}^{J} k_j(P_j t, P_j t') + \sigma^2 \delta_{t,t'} \,, \tag{1.4.12}$$

[12] The widely used RBF or Gaussian kernel is an example of a tensor kernel.

where $P_j : \mathbb{R}^d \to \mathbb{R}$ projection and each sub-kernel k_j is parameterized by θ_j in some family of kernels [52]. This PPGPR has linear computational complexity using an algorithm inspired by backfitting.

A recent work [26] avoids this pursuit of optimal projections by showing the convergence of kernel $\frac{1}{J}\sum_{j=1}^{J} k(P_j t, P_j t')$, for randomly selected projections $P_j : \mathbb{R}^d \to \mathbb{R}$ and scalar input translationally invariant kernel k, to a limiting kernel. It is also shown that a relatively small number of random projections in additive kernel

$$k_J(t,t') = \sum_{j=1}^{J} \alpha_j k_j(P_j t, P_j t') \tag{1.4.13}$$

are needed to obtain numerical convergence in GPR classification error. This leads to efficient algorithms using the previously mentioned SKI trick.

1.5 Introduction to pattern learning problems

We have summarized the theory and applications of Gaussian process regression and kernel interpolation (more in depth exposition can be found in [109, 113]). Next, we will present the background to three problems that we will approach with GPR in this thesis. We will also exhibit the need to learn patterns in data to effectively address each problem.

Denoising solutions to linear equations

The context of the denoising of linear equation solutions, as in [153] and further presented in Section 2, will be introduced using the following problem.

Problem 2. *Suppose* $\Omega \subset \mathbb{R}^d$ *(*$d \in \mathbb{N}$*) is a regular bounded domain,* $\mathcal{L} : \mathcal{H}_0^s(\Omega) \to \mathcal{H}^{-s}(\Omega)$ *is a symmetric positive local*[13] *linear bijection, and* $\|f\|^2 := \int_\Omega f \mathcal{L} f$ *is the energy-norm associated with* $\mathcal{L}$*. It is assumed that* u *is such that* $\mathcal{L}u \in L^2(\Omega)$ *with* $\|\mathcal{L}u\|_{L^2(\Omega)} \leq M$ *and that* $\zeta \sim \mathcal{N}(0, \sigma\delta(x-y))$*. Given the noisy observation* $\eta = u + \zeta$*, find an approximation of* u *that is as accurate as possible in the energy norm* $\|\cdot\|$*.*

To illustrate this problem, we consider elliptic partial differential operator $\mathcal{L} = -\operatorname{div}\big(a(x)\nabla\,\cdot\,\big)$ with $\Omega = [0,1] \subset \mathbb{R}^1$. Hence, $\mathcal{L}u = g \in L^2(\Omega)$ is a partial differential equation (PDE). In Figure 1.1.1, we show a fixed example of a. In

[13] Symmetric positive local is defined as $\int_\Omega u\mathcal{L}v = \int_\Omega v\mathcal{L}u$, $\int_\Omega u\mathcal{L}u > 0$ for $u \neq 0$, and $\int_\Omega u\mathcal{L}v = 0$ for $u, v \in \mathcal{H}_0^s(\Omega)$ with disjoint supports, respectively.

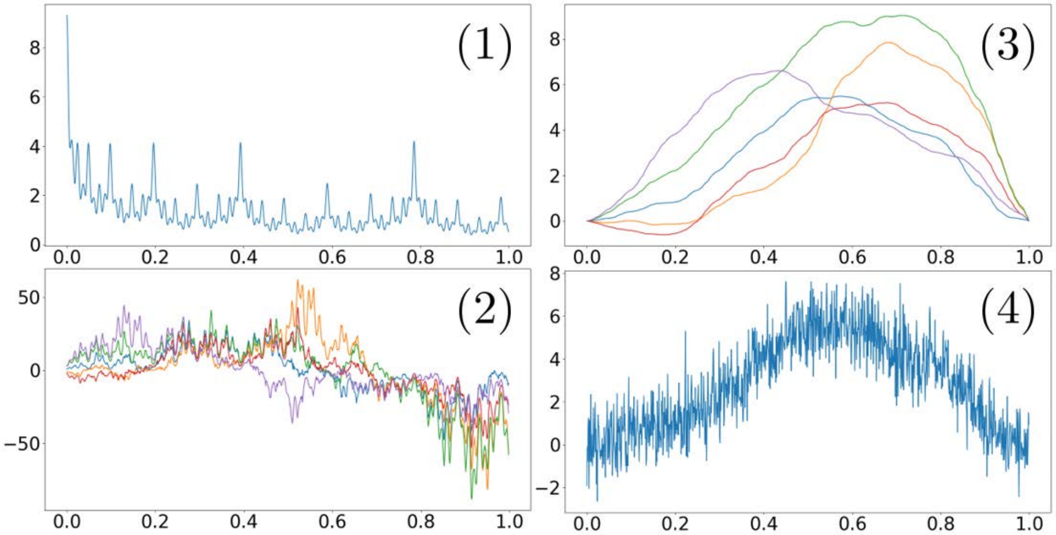

Figure 1.1: Illustrations showing (1) a (2) 5 examples of ∇u (3) 5 examples of u (4) 1 example of $\eta = u + \zeta$.

Figures 1.1.2–3, we show 5 examples of u for random realizations of g. We see that while global characteristics of the examples differ, local patterns, i.e., maxima and inflection points in ∇u and u, respectively, are located in identical locations. As expected, these locations are determined by the coefficients a in PDE $\mathcal{L}u = g$. This denoising problem observes $\eta = u + \zeta$, e. g., Figure 1.1.4. The local characteristics of u are lost in η due to noise, and the problem requires an accurate estimation of these patterns. Section 2 will present a result that shows that Gaussian process conditioning yields a near minimax recovery, i.e., one that is within a fixed constant of a minimax recovery in norm $\|\cdot\|$. We also numerically observe the lost patterns recovered.

Mode decomposition

We next present the mode decomposition problem detailed in Sections 3, 4, and 5. This problem is summarized by the following statement.

Problem 3. *For $m \in \mathbb{N}$, let $a_1, \ldots, a_m$ be piecewise smooth functions on interval $I \subset \mathbb{R}$, and let $\theta_1, \ldots, \theta_m$ be strictly increasing functions on I. Assume that m and the a_i, θ_i are unknown. Given the observation of $v(t) = \sum_{i=1}^m a_i(t) \cos\big(\theta_i(t)\big), t \in I$, recover the modes $v_i(t) := a_i(t) \cos\big(\theta_i(t)\big)$.*

Modes v_i defined in this manner are called *nearly periodic*, and they must each be estimated from the composite signal v which loses the patterns of each mode. This

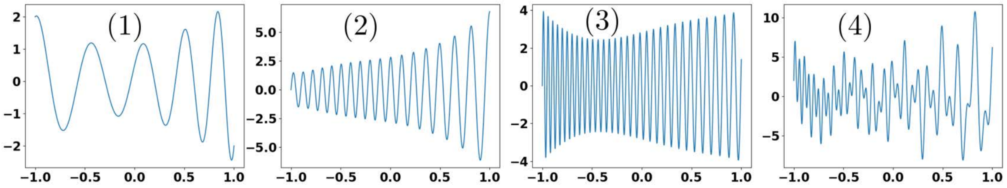

Figure 1.2: Illustrations showing (1) v_1 (2) v_2 (3) v_3 (4) $v = v_1 + v_2 + v_3$ in an example of Problem 3.

problem is made more challenging in Section 4, where we generalize the base cosine waveform, i.e., defining $v(t) = \sum_{i=1}^{m} a_i(t) y\big(\theta_i(t)\big)$ for some known square integrable periodic y.

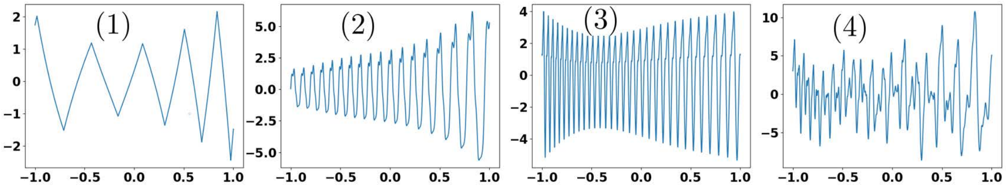

Figure 1.3: Illustrations showing (1) v_1 (2) v_2 (3) v_3 (4) $v = v_1 + v_2 + v_3$ in an example of the variant of Problem 3 with arbitrary unknown waveforms.

This is further generalized in Section 5 to mode recovery when each waveform is unknown, i.e., defining $v(t) = \sum_{i=1}^{m} a_i(t) y_i\big(\theta_i(t)\big)$ for unknown square integrable periodic y_i. Figure 1.3 illustrates signal v composed of modes v_1, v_2, and v_3 with arbitrary waveforms. The waveform patterns of each mode are mixed and become difficult to discern in v. Additionally, mode decomposition with the presence of noise in the observed signal $v(t)$ is addressed in [102, Sec. 10].

Image classification

We describe the image classification problem, which is a canonical example of machine learning and artificial intelligence. A classification problem, which is a form of supervised learning, is informally given in the following statement.

Problem 4. *Suppose the domain $\Omega \subset \mathbb{R}^N$ is divided into multiple classes. Given training data (x_i, y_i) where $x_i \in \Omega$ and y_i is the class of x_i, learn a classifier of Ω, i.e., a mapping from all $x \in \Omega$ to class $y(x)$, that is accurate on testing data (x_i^t, y_i^t).*

In the image learning problem, the data of each image consists of a grid of pixels. In the case of grayscale images, each pixel takes a value in $\mathbb{R}$ representing brightness at each position. Overall the image can be expressed equivalently in matrix or vector form, i.e., $\mathbb{R}^{m\times n}$ or $\mathbb{R}^{mn}$, respectively. However, in RGB color images, each pixel represents the brightness of each of the red, green, and blue components and hence each image is an element of $\mathbb{R}^{3\times m\times n}$ or $\mathbb{R}^{3mn}$.

While it is unknown exactly how the human brain is able to understand the content of images, it is more naturally suited to recognize patterns than a computer, which only observes the vector corresponding to brightness at each pixel. The aim of the image classification problem is to automate this human pattern recognition by constructing a map from high dimensional image space to class. Only recently in the 2010s have accuracy rates of machines exceeded those of humans [61, 64], showing the difficulty of this problem.

We present four datasets with training images sampled in Figure 1.4. The top row shows examples from the MNIST dataset [83], which consists of 60000 training and 10000 testing grayscale images of written digits 0 through 9. Each image is of size 28×28 and classified according to digit into one of the 10 classes. The middle row shows examples from fashion MNIST [144], which, identically to MNIST, consists of 60000 training and 10000 testing 28×28 grayscale images. These images, however, are of articles of clothing, divided into 10 classes: t-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. The CIFAR-10 and CIFAR-100 datasets [80] both have 50000 training and 10000 testing 32×32 RGB images. The CIFAR-10 and CIFAR-100 datasets differ in having 10 and 100 image classes, respectively. The CIFAR-10 classes are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. A list of the CIFAR-100 classes can be found in [80, Appx. D].

Both MNIST and fashion MNIST images are expressible as $\mathbb{R}^{784}$ vectors. The CIFAR-10 and CIFAR-100 images are expressible as $\mathbb{R}^{3072}$ vectors. A well known challenge in learning and classification problems over high dimensional spaces is that an exponentially large training dataset is needed in a dense sampling. For example, in $\Omega = [0, 1]^n$, approximately 10^n points are needed to sample Ω such that each point is at most 0.1 units away[14] from the nearest other point. This issue is known as the *curse of dimensionality*. Learning is made possible in MNIST, CIFAR, and

[14] Distance measured with the Euclidean metric

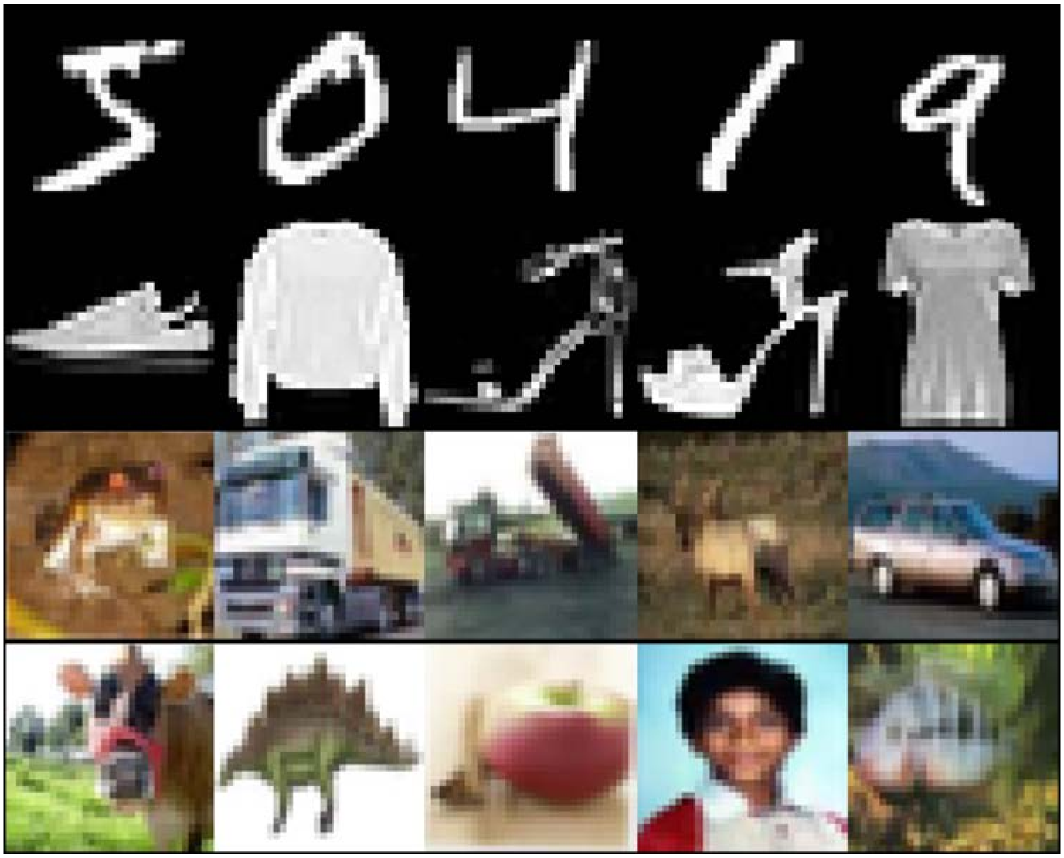

Figure 1.4: The top row shows representatives from the MNIST dataset with images in the classes 5, 0, 4, 1, and 9, from left to right. The second row shows fashion MNIST images from the classes sneaker, pullover, sandal, sandal, and T-shirt/top. The third row shows CIFAR-10 images from the classes frog, truck, truck, deer, and automobile. The bottom row shows CIFAR-100 images from the classes cattle, dinosaur, apple, boy, and aquarium fish.

other real-world image datasets because the set of images in each class is a relatively low-dimensional space embedded in the whole image space. Indeed, a randomly selected image in the MNIST or CIFAR image spaces will almost always appear as noise and has an infinitesimal probability of containing the defining patterns and characteristics of a particular image class, such as a sandal or automobile. The digit MNIST and CIFAR-10 datasets have been shown to have effective dimensions of approximately 15 and 35 [123]. Further, the dimensions of the spaces of individual classes in digit MNIST have been estimated to be between 8 and 13 [21, 116]. We show how incorporating kernel techniques into conventional image classification methods can improve accuracy rates in Section 7.

1.6 Patterns and kernels

As discussed in Section 1.3, when applying GPR to general regression problems, the selection of model GP[15] greatly influences the result. An appropriate covariance kernel which models the patterns of the data must be chosen. The conventional approach is to make a selection based on underlying assumptions. This chosen kernel is then used to make a predictor incorporating both the training data and assumptions of covariance.

This thesis will present novel approaches to the pattern learning problems discussed in Section 1.5. In the denoising and mode decomposition problems, we use a kernel inspired from assumptions in the respective problems. Then, we will estimate patterns of the PDE solution $\mathcal{L}u = g \in L^2(\Omega)$ and of each mode v_i in signal $v = v_1 + \cdots + v_m$ with GPR along side respective conventional techniques. We will further elaborate *learning patterns with kernels* after this paragraph. In the image learning problem, there are no assumptions on the structure of the high-dimensional image vectors. A covariance kernel that effectively models image data is a well known challenge. We present the *Kernel Flow* (KF) algorithm at the end of this section as a method for data-based kernel selection, in other words, for *learning kernels from patterns*.

Learning patterns with kernels

The first main theme of this work will be showcased using the denoising and mode decomposition problems. In both of these contexts, we overview how GPR using kernels constructed from the assumptions of each problem is able to recover underlying patterns. As illustrated in Figures 1.1, 1.2, and 1.3, these patterns are learned from data where they are visually incomprehensible due to corruptions from either noise or other modes.

In the denoising problem, we utilize the operator-adapted wavelet, also known as gamblet, transform [153]. These hierarchical wavelets are defined to be associated with a particular operator, and in the context of Problem 2, we choose $\mathcal{L}$-adapted wavelets. Exposition on gamblets will be given in Section 2.2, and further details can be found in [101]. Theorem 2.2.4 presents the result that these gamblets are precisely the conditional expectation of the canonical Gaussian field[16] corresponding to $\mathcal{L}$ with respect to hierarchical measurement functions. A truncation of the gamblet

[15] Or equivalently the selection of kernel.

[16] A Gaussian field is a generalization of a Gaussian process in Banach Spaces. Gaussian fields are GPs when point-wise evaluation is continuous. Further details can be found in Def. 2.2.1.

transform described in Theorem 2.3.1 is found to be within a constant of a minimax optimal recovery. This denoising is interpretable as a Gaussian conditioning with covariance kernel derived from $\mathcal{L}$. The length-scale of the conditioning is dependent on the signal-to-noise ratio in the assumption of the problem. A conventional method for denoising smooth signals involves thresholding empirical wavelet coefficients. We numerically compare the near-minimax optimal recovery with thresholding gamblet transform coefficients in Section 2.4.

Our approach [102] to the mode decomposition problem, kernel mode decomposition (KMD), is outlined in Section 4. There are two main components within the algorithm: The first, which we name max-pooling, estimates the instantaneous phase and frequency of the lowest frequency mode in a signal and is presented in Section 4.1. It is a close variant of the continuous wavelet transform (CWT) [24], which we summarize in Section 3.2. The second component uses GPR to estimate the instantaneous amplitude and phase of this lowest frequency mode and is summarized in Section 4.2. We define a GP with a covariance kernel constructed from Gaussian-windowed trigonometric waves, i.e., Gabor wavelets [48]. GPR then is able to estimate the instantaneous phase and amplitude of the lowest frequency mode. Further, when the base waveform, i.e., y_i in $v(t) = \sum a_i(t)y_i(\theta_i(t))$, of each mode is unknown, GPR can be applied to estimate y_i as presented in Section 5.1. These algorithms were extended in [102, Sec. 10], showing the method can be constructed to be robust to noise, vanishing amplitudes, and modes with crossing frequencies. We find that patterns within each mode can be estimated with GPR from the sum of modes, even when these patterns are visually indistinguishable in the composite signal.

Learning kernels from patterns

We will discuss the Kernel Flow (KF) algorithm [154] next. At a high level, the algorithm is interpretable as learning kernels from patterns with a method inspired by cross-validation. The algorithm is described in Section 6 and operates under the principle that a kernel is desirable when it can make low error predictions with small samples of the whole data set. This error is quantified by selecting N random training points and computing the kernel interpolation with a further random $N/2$ of these points. We compute the error of the of the interpolation on the other $N/2$ points. Assuming these interpolations are written as $u^{\dagger,f}$ and $u^{\dagger,c}$ respectively, the error is quantified as the KF loss function,

$$\rho := \frac{\|u^{\dagger,f} - u^{\dagger,c}\|_{k_\theta}^2}{\|u^{\dagger,f}\|_{k_\theta}^2}, \tag{1.6.1}$$

and the KF algorithm selects a kernel with optimization. Since this loss function is dependent mainly on the training data, this method selects a kernel based on patterns in that data. We present examples throughout Section 6 including on the MNIST dataset. We find that this technique is able to learn kernels which can predict classes accurately only observing one point per class. Additionally, we observe evidence of unsupervised learning since archetypes within each class appear to be learned. A further example of pattern learning applying the KF algorithm can be found in [57], where it has been applied to data in chaotic dynamical systems to learn a model kernel.

An application of the KF algorithm to Artificial Neural Networks (ANNs) will further be demonstrated in Section 7 to improve key performance statistics in MNIST and CIFAR image classification problems. These ANNs are widely used to address this problem and are defined as the mapping

$$f_\theta(x) = \left(f_{\theta_n}^{(n)} \circ f_{\theta_{n-1}}^{(n-1)} \circ \cdots \circ f_{\theta_1}^{(1)}\right)(x). \tag{1.6.2}$$

This map has input x and n layers $f_{\theta_i}^{(i)}(z) = \phi(W_i z + b_i)$ parameterized[17] by the weights and biases $\theta_i := (W_i, b_i)$, $\theta := \{\theta_1, \ldots, \theta_n\}$. The output of f_θ is in $\mathbb{R}^c$, where c represents the number of classes in the dataset. This is converted into a classifier by selecting the component with largest value. The parameters θ best modeling the patterns of the data are learned by optimizing the error of the classifier, usually with cross-entropy loss[18], on the training data.

Kernels can be incorporated into ANNs by allowing f_θ to map into a higher dimensional space and applying kernel interpolation on the result, which leads to an improvement of error rates [103, Sec. 10]. Further improvements can be made by reverting to the standard f_θ largest component classifier and constructing a kernel dependent on intermediate-layer output

$$h^{(i)}(x) := \left(f_{\theta_i}^{(i)} \circ f_{\theta_{i-1}}^{(i-1)} \circ \cdots \circ f_{\theta_1}^{(1)}\right)(x), \tag{1.6.3}$$

for $i = 1, \ldots, n$. The KF loss corresponding to this kernel is then used in tandem with the standard cross-entropy loss, which leads to improvements in testing error,

[17] Weights, W_i, are linear operators and biases, b_i are vectors. The function ϕ is an arbitrary function, typically taken as the ReLU, $\phi(z) = \max(0, z)$.

[18] This loss is defined in equation (7.0.4).

generalization gap, and robustness to distributional shift. Details on our numerical findings can be found in Section 7.1. Note that kernel interpolation itself is not directly used as a classifier; the KF loss is used only as a regularization of the loss function used in the optimization of the ANN parameters. This application of kernels is a novel method for training and clustering intermediate-layer outputs in conjunction with the final output f_θ. We present numerical experiments that show the KF loss function aids in the learning of parameters which most accurately classify patterns in images.

Chapter 2

DENOISING

2.1 Introduction to the denoising problem

[37–39] addressed the problem of recovering of a smooth signal from noisy observations by soft-thresholding empirical wavelet coefficients [37]. More recently, [33] considered the recovery of $x \in X$ based on the observation of $Tx+\zeta$, where ζ_i is IID. $\mathcal{N}(0,\sigma^2)$ and T is a compact linear operator between Hilbert spaces X and Y, with the prior that x lies in an ellipsoid defined by the eigenvectors of T^*T. [33] showed that thresholding the coefficients of the corrupted signal $Tx+\zeta$ in the basis formed by the singular value decomposition (SVD) of T (which can be computed in (N^3) complexity) approached the minimax recovery to a fixed multiplicative constant.

The contributions presented in this section [153] address denoisings in the following formulation. Suppose

$$\mathcal{L} : \mathcal{H}^s_0(\Omega) \to \mathcal{H}^{-s}(\Omega) \tag{2.1.1}$$

is a symmetric positive local[1] linear bijection with $s \in \mathbb{N}^*$ and regular bounded $\Omega \subset \mathbb{R}^d$ ($d \in \mathbb{N}$). Let $\|\cdot\|$ be the energy-norm defined by

$$\|u\|^2 := \int_\Omega u\mathcal{L}u\,, \tag{2.1.2}$$

and write

$$\langle u, v\rangle := \int_\Omega u\mathcal{L}v \tag{2.1.3}$$

for the associated scalar product. Further, define

$$V_M := \{u \in \mathcal{H}^s_0(\Omega) : \mathcal{L}u \in L^2(\Omega) \text{ and } \|\mathcal{L}u\|_{L^2(\Omega)} \leq M\}\,. \tag{2.1.4}$$

Further, let

$$\zeta \sim \mathcal{N}(0, \sigma^2\delta(x-y)) \tag{2.1.5}$$

be white noise in domain Ω with variance σ^2. The following is the continuous version of the denoising problem studied in this section.

Problem 5. *Let u be an unknown element of V_M, given the noisy observation $\eta = u+\zeta$, find an approximation of u that is as accurate as possible in the energy norm $\|\cdot\|$.*

[1] Symmetric positive local defined as $\int_\Omega u\mathcal{L}v = \int_\Omega v\mathcal{L}u$, $\int_\Omega u\mathcal{L}u > 0$ for $u \neq 0$, and $\int_\Omega u\mathcal{L}v = 0$ for $u, v \in \mathcal{H}^s_0(\Omega)$ with disjoint supports, respectively.

This problem will be illustrated in the $s = 1$ case with the linear differential operator $\mathcal{L} = -\operatorname{div}\left(a(x)\nabla\cdot\right)$ where the conductivity a is a uniformly elliptic symmetric $d \times d$ matrix with entries in $L^\infty(\Omega)$. This example is of practical importance in groundwater flow modeling (where a is the porosity of the medium) and in electrostatics (where a is the dielectric constant), and in both applications a may be rough (non-smooth) [7, 19].

Example 2.1.1. *Assuming*

$$\mathcal{L} = -\operatorname{div}\left(a(x)\nabla\cdot\right) : \mathcal{H}_0^1(\Omega) \to \mathcal{H}^{-1}(\Omega)\,, \tag{2.1.6}$$

Prob. 5 then corresponds to the problem of recovering the solution of the PDE

$$\begin{cases} -\operatorname{div}\left(a(x)\nabla u(x)\right) = f(x) & x \in \Omega; \\ u = 0 & \text{on} \quad \partial\Omega, \end{cases} \tag{2.1.7}$$

from its noisy observation $\eta = u + \zeta$ with knowledge $\|f\|_{L^2(\Omega)} < M$.

This problem is addressed by expressing η in the gamblet transform adapted to operator $\mathcal{L}$ and applying a truncation to the series. This method is theoretically proved to yield a recovery within a constant of the minimax optimal recovery [153]. This method is numerically compared to thresholding the gamblet transform coefficients as well as *regularization*, the minimization of

$$\|v(\eta) - \eta\|^2_{L^2(\Omega)} + \alpha\|v(\eta)\|^2\,. \tag{2.1.8}$$

2.2 Summary of operator-adapted wavelets

We proceed by reviewing *operator-adapted wavelets* as in [153, Sec. 2], also named *gamblets* in reference to their game theoretic interpretation, and their main properties [99, 101, 104, 119]. They are constructed with a hierarchy of measurement functions and an operator. Theorem 2.2.4 shows these gamblets are simultaneously associated with Gaussian conditioning, optimal recovery, and game theory. By selecting these measurement functions to be *pre-Haar wavelets*, the gamblets are localized both in space and in the eigenspace of the operator.

Hierarchy of measurement functions

Let $q \in \mathbb{N}^*$ (used to represent a number of scales). Let $(\mathcal{I}^{(k)})_{1\le k\le q}$ be a hierarchy of labels defined as follows. $\mathcal{I}^{(q)}$ is a set of q-tuples consisting of elements $i = (i_1, \ldots, i_q)$. For $1 \le k \le q$ and $i \in \mathcal{I}^{(q)}$, $i^{(k)} := (i_1, \ldots, i_k)$ and $\mathcal{I}^{(k)}$ is the set of

k-tuples $\mathcal{I}^{(k)} = \{i^{(k)} | i \in \mathcal{I}^{(q)}\}$. For $1 \leq r \leq k \leq q$ and $j = (j_1, \dots, j_k) \in \mathcal{I}^{(k)}$, we write $j^{(r)} = (j_1, \dots, j_r)$. We say that M is a $\mathcal{I}^{(k)} \times \mathcal{I}^{(l)}$ matrix if its rows and columns are indexed by elements of $\mathcal{I}^{(k)}$ and $\mathcal{I}^{(l)}$, respectively.

Let $\{\phi_i^{(k)} | k \in \{1, \dots, q\},\, i \in \mathcal{I}^{(k)}\}$ be a nested hierarchy of elements of $\mathcal{H}^{-s}(\Omega)$ such that $(\phi_i^{(q)})_{i \in \mathcal{I}^{(q)}}$ are linearly independent and

$$\phi_i^{(k)} = \sum_{j \in \mathcal{I}^{(k+1)}} \pi_{i,j}^{(k,k+1)} \phi_j^{(k+1)} \tag{2.2.1}$$

for $i \in \mathcal{I}^{(k)}$, $k \in \{1, \dots, q-1\}$, where $\pi^{(k,k+1)}$ is an $\mathcal{I}^{(k)} \times \mathcal{I}^{(k+1)}$ matrix and

$$\pi^{(k,k+1)} \pi^{(k+1,k)} = I^{(k)} \,. \tag{2.2.2}$$

In (2.2.2), $\pi^{(k+1,k)}$ is the transpose of $\pi^{(k,k+1)}$ and $I^{(k)}$ is the $\mathcal{I}^{(k)} \times \mathcal{I}^{(k)}$ identity matrix.

Hierarchy of operator-adapted pre-wavelets

Let $(\psi_i^{(k)})_{i \in \mathcal{I}^{(k)}}$ be the hierarchy of optimal recovery splines associated with $(\phi_i^{(k)})_{i \in \mathcal{I}^{(k)}}$, i.e., for $k \in \{1, \dots, q\}$ and $i \in \mathcal{I}^{(k)}$,

$$\psi_i^{(k)} = \sum_{j \in \mathcal{I}^{(k)}} A_{i,j}^{(k)} \mathcal{L}^{-1} \phi_j^{(k)} \,, \tag{2.2.3}$$

where

$$A^{(k)} := (\Theta^{(k)})^{-1} \tag{2.2.4}$$

and $\Theta^{(k)}$ is the $\mathcal{I}^{(k)} \times \mathcal{I}^{(k)}$ symmetric positive definite Gramian matrix with entries (writing $[\phi, v]$ for the duality pairing between $\phi \in \mathcal{H}^{-s}(\Omega)$ and $v \in \mathcal{H}_0^s(\Omega)$)

$$\Theta_{i,j}^{(k)} = [\phi_i^{(k)}, \mathcal{L}^{-1} \phi_j^{(k)}] \,. \tag{2.2.5}$$

Note that $A^{(k)}$ is the stiffness matrix of the elements $(\psi_i^{(k)})_{i \in \mathcal{I}^{(k)}}$ in the sense that

$$A_{i,j}^{(k)} = \langle \psi_i^{(k)}, \psi_j^{(k)} \rangle. \tag{2.2.6}$$

Writing $\Phi^{(k)} := \operatorname{span}\{\phi_i^{(k)} \mid i \in \mathcal{I}^{(k)}\}$ and $\mathfrak{B}^{(k)} := \operatorname{span}\{\psi_i^{(k)} \mid i \in \mathcal{I}^{(k)}\}$, $\Phi^{(k)} \subset \Phi^{(k+1)}$ and $\Psi^{(k)} = \mathcal{L}^{-1} \Phi^{(k)}$ imply $\Psi^{(k)} \subset \Psi^{(k+1)}$. We further write $[\phi^{(k)}, u] = \big([\phi_i^{(k)}, u]\big)_{i \in \mathcal{I}^{(k)}} \in \mathbb{R}^{\mathcal{I}^{(k)}}$.

The $(\phi_i^{(k)})_{i \in \mathcal{I}^{(k)}}$ and $(\psi_i^{(k)})_{i \in \mathcal{I}^{(k)}}$ form a bi-orthogonal system in the sense that

$$[\phi_i^{(k)}, \psi_j^{(k)}] = \delta_{i,j} \text{ for } i, j \in \mathcal{I}^{(k)} \tag{2.2.7}$$

and the $\langle \cdot, \cdot \rangle$-orthogonal projection of $u \in \mathcal{H}_0^s(\Omega)$ on $\Psi^{(k)}$ is

$$u^{(k)} := \sum_{i \in \mathcal{I}^{(k)}} [\phi_i^{(k)}, u] \psi_i^{(k)} \,. \tag{2.2.8}$$

Multiple interpretations of operator adapted pre-wavelets

Using operator-adapted pre-wavelets, $\psi_i^{(k)}$, we summarize the connections between optimal recovery, game theory, and Gaussian conditioning. First, we define Gaussian fields, a generalization of Gaussian processes.

Definition 2.2.1. *The canonical Gaussian field ξ associated with operator $\mathcal{L} : \mathcal{H}_0^s(\Omega) \to \mathcal{H}^{-s}(\Omega)$ is defined such that $\phi \mapsto [\phi, \xi]$ is the linear isometry from $\mathcal{H}^{-s}(\Omega)$ to a Gaussian space characterized by*

$$\begin{cases} [\phi, \xi] \sim \mathcal{N}(0, \|\phi\|_*^2) \\ \operatorname{Cov}\big([\phi, \xi], [\varphi, \xi]\big) = \langle \phi, \varphi \rangle_* , \end{cases} \tag{2.2.9}$$

where $\|\phi\|_* = \sup_{u \in \mathcal{H}_0^s(\Omega)} \frac{\int_\Omega \phi u}{\|u\|}$ is the dual norm of $\| \cdot \|$.

Remark 2.2.2. *When $s > d/2$, the evaluation functional $\delta_x(f) = f(x)$ is continuous. Hence, $\xi|_{\delta_x, x \in \Omega}$ is naturally isomorphic to a Gaussian process with covariance function $k(x, x') = \langle \delta_x, \delta_{x'} \rangle_*$.*

Several notable properties of these pre-wavelets are summarized in the following result. Recall we write $[\phi^{(k)}, u] = \big([\phi_i^{(k)}, u]\big)_{i \in \mathcal{I}^{(k)}} \in \mathbb{R}^{\mathcal{I}^{(k)}}$.

Theorem 2.2.3. *Consider pre-wavelets $\psi_i^{(k)}$ adapted to operator $\mathcal{L}$ constructed with measurement functions $\phi_i^{(k)}$. Further, suppose that for $u \in \mathcal{H}_0^s(\Omega)$ we define $v^\dagger(u) = u^{(k)} = \sum_{i \in \mathcal{I}^{(k)}} [\phi_i^{(k)}, u] \psi_i^{(k)}$.*

1. *For fixed $u \in \mathcal{H}_0^s(\Omega)$, $v^\dagger(u)$ is the minimizer of*

$$\begin{cases} \text{Minimize } \|\psi\| \\ \text{Subject to } \psi \in \mathcal{H}_0^s(\Omega) \text{ and } [\phi^{(k)}, \psi] = [\phi^{(k)}, u] . \end{cases} \tag{2.2.10}$$

2. *For fixed $u \in \mathcal{H}_0^s(\Omega)$, $v^\dagger(u)$ is the minimizer of*

$$\begin{cases} \text{Minimize } \|u - \psi\| \\ \text{Subject to } \psi \in \operatorname{span}\{\psi_i^{(k)} : i \in \mathcal{I}^{(k)}\} . \end{cases} \tag{2.2.11}$$

3. *For canonical Gaussian field $\xi \sim \mathcal{N}(0, \mathcal{L}^{-1})$,*

$$v^\dagger(u) = \mathbb{E}\big[\xi \big| [\phi^{(k)}, \xi] = [\phi^{(k)}, u]\big] . \tag{2.2.12}$$

4. *It is true that*[2]

$$v^{\dagger} \in \operatorname{argmin}_{v \in L(\Phi, \mathcal{H}_0^s(\Omega))} \sup_{u \in \mathcal{H}_0^s(\Omega)} \frac{\|u - v(u)\|}{\|u\|} \tag{2.2.13}$$

Proof. (1) is a result of [101, Cor. 3.4] (2) is equivalent to [101, Thm. 12.2] (3) and (4) are results in [101, Sec. 8.5]. □

This result shows that the operator-adapted pre-wavelets transform defined by $v^{\dagger}(u) = u^{(k)}$ is an optimal recovery in the sense of Theorem 2.2.3.1-2. Simultaneously, $v^{\dagger}(u)$ are conditional expectations of the canonical Gaussian field with respect to the measurements $[\phi^{(k)}, \cdot]$ as in Theorem 2.2.3.3. Another interpretation of the transform is game theoretic as expressed in Theorem 2.2.13.4. Equation (2.2.13) represents the adversarial two player game where player I selects $u \in \mathcal{H}_0^s(\Omega)$ and player II approximates u with $v(u)$ with measurements $[\phi^{(k)}, u]$. Player I and II aim to maximize and minimize the recovery error of $v(u)$. This game theoretic interpretation inspires the name *gamblets*, referring to operator-adapted wavelets. Note that the pre-wavelets $\psi_i^{(k)}$ lie on only one level of the hierarchy. The following addresses the construction of a wavelet decomposition of $\mathcal{H}_0^s(\Omega)$ on all hierarchical levels.

Operator-adapted wavelets

Let $(\mathcal{J}^{(k)})_{2 \le k \le q}$ be a hierarchy of labels such that, writing $|\mathcal{J}^{(k)}|$ for the cardinal of $\mathcal{J}^{(k)}$,

$$|\mathcal{J}^{(k)}| = |\mathcal{I}^{(k)}| - |\mathcal{I}^{(k-1)}| \,. \tag{2.2.14}$$

For $k \in \{2, \dots, q\}$, let $W^{(k)}$ be a $\mathcal{J}^{(k)} \times \mathcal{I}^{(k)}$ matrix such that[3]

$$\operatorname{Ker}(\pi^{(k-1,k)}) = \operatorname{Im}(W^{(k),T}) \,. \tag{2.2.15}$$

For $k \in \{2, \dots, q\}$ and $i \in \mathcal{J}^{(k)}$ define

$$\chi_i^{(k)} := \sum_{j \in \mathcal{I}^{(k)}} W_{i,j}^{(k)} \psi_j^{(k)} \,, \tag{2.2.16}$$

and write $\mathfrak{W}^{(k)} := \operatorname{span}\{\chi_i^{(k)} \mid i \in \mathcal{J}^{(k)}\}$. Then $\mathfrak{W}^{(k)}$ is the $\langle \cdot, \cdot \rangle$-orthogonal complement of $\mathfrak{V}^{(k-1)}$ in $\mathfrak{V}^{(k)}$, i.e. $\mathfrak{V}^{(k)} = \mathfrak{V}^{(k-1)} \oplus \mathfrak{W}^{(k)}$, and

$$\mathfrak{V}^{(q)} = \mathfrak{V}^{(1)} \oplus \mathfrak{W}^{(2)} \oplus \cdots \oplus \mathfrak{W}^{(q)} \,. \tag{2.2.17}$$

[2] $L(\Phi, \mathcal{H}_0^s(\Omega))$ is defined as the set of $\mathcal{H}_0^s(\Omega) \to \mathcal{H}_0^s(\Omega)$ functions that are of form $v(u) = \Psi\big([\phi^{(k)}, u]\big)$ with measureable $\Psi : \mathbb{R}^{\mathcal{I}^{(k)}} \to \mathcal{H}_0^s(\Omega)$.

[3] We write $M^{(k),T}$ and $M^{(k),-1}$ for the transpose and inverse of a matrix $M^{(k)}$.

For $k \in \{2,\ldots,q\}$ write

$$B^{(k)} := W^{(k)} A^{(k)} W^{(k),T}\,. \tag{2.2.18}$$

Note that $B^{(k)}$ is the stiffness matrix of the elements $(\chi_j^{(k)})_{j\in\mathcal{J}^{(k)}}$, i.e.,

$$B_{i,j}^{(k)} = \langle \chi_i^{(k)}, \chi_j^{(k)} \rangle\,. \tag{2.2.19}$$

Further, for $k \in \{2,\ldots,q\}$, define

$$N^{(k)} := A^{(k)} W^{(k),T} B^{(k),-1} \tag{2.2.20}$$

and, for $i \in \mathcal{J}^{(k)}$,

$$\phi_i^{(k),\chi} := \sum_{j\in\mathcal{I}^{(k)}} N_{i,j}^{(k),T} \phi_j^{(k)}\,. \tag{2.2.21}$$

Then defining $u^{(k)}$ as in (2.2.8), it holds true that for $k \in \{2,\ldots,q\}$, $u^{(k)} - u^{(k-1)}$ is the $\langle\cdot,\cdot\rangle$-orthogonal projection of u on $\mathfrak{W}^{(k)}$ and

$$u^{(k)} - u^{(k-1)} = \sum_{i\in\mathcal{J}^{(k)}} [\phi_i^{(k),\chi}, u]\chi_i^{(k)}\,. \tag{2.2.22}$$

To simplify notations, write $\mathcal{J}^{(1)} := \mathcal{I}^{(1)}$, $B^{(1)} := A^{(1)}$, $N^{(1)} := I^{(1)}$, $\phi_i^{(1),\chi} := \phi_i^{(1)}$ for $i \in \mathcal{J}^{(1)}$, $\mathcal{J} := \mathcal{J}^{(1)} \cup \cdots \cup \mathcal{J}^{(q)}$, $\chi_i := \chi_i^{(k)}$ and $\phi_i^{\chi} := \phi_i^{(k),\chi}$ for $i \in \mathcal{J}^{(k)}$ and[4] $1 \le k \le q$. Then the ϕ_i^{χ} and χ_i form a bi-orthogonal system, i.e.,

$$[\phi_i^{\chi}, \chi_j] = \delta_{i,j} \text{ for } i,j \in \mathcal{J} \tag{2.2.23}$$

and

$$u^{(q)} = \sum_{i\in\mathcal{J}} [\phi_i^{\chi}, u]\chi_i\,. \tag{2.2.24}$$

Simplifying notations further, we will write $[\phi^{\chi}, u]$ for the $\mathcal{J}$ vector with entries $[\phi_i^{\chi}, u]$ and χ for the $\mathcal{J}$ vector with entries χ_i so that (2.2.24) can be written

$$u^{(q)} = [\phi^{\chi}, u] \cdot \chi\,. \tag{2.2.25}$$

Further, define the $\mathcal{J}$ by $\mathcal{J}$ block-diagonal matrix B defined as $B_{i,j} = B_{i,j}^{(k)}$ if $i,j \in \mathcal{J}^{(k)}$ and $B_{i,j} = 0$ otherwise. Note that it holds that $B_{i,j} = \langle \chi_i, \chi_j \rangle$. When $q = \infty$ and $\cup_{k=1}^{\infty} \Phi^{(k)}$ is dense in $\mathcal{H}^{-s}(\Omega)$, then, writing $\mathfrak{W}^{(1)} := \mathfrak{V}^{(1)}$,

$$\mathcal{H}_0^s(\Omega) = \oplus_{k=1}^{\infty} \mathfrak{W}^{(k)}\,, \tag{2.2.26}$$

[4] The dependence on k is left implicit to simplify notation, for $i \in \mathcal{J}$ there exists a unique k such that $i \in \mathcal{J}^{(k)}$.

$u^{(q)} = u$, and (2.2.24) is the corresponding multi-resolution decomposition of u. When $q < \infty$, $u^{(q)}$ is the projection of u on $\oplus_{k=1}^{q} \mathfrak{W}^{(k)}$ and (2.2.25) is the corresponding multi-resolution decomposition. Note that the optimal recovery, game theory, and Gaussian conditioning results in Theorem 2.2.3 also holds for wavelets.

Theorem 2.2.4. *Consider pre-wavelets χ_i adapted to operator $\mathcal{L}$ constructed with measurement functions ϕ^{χ}. Further, suppose that for $u \in \mathcal{H}_0^s(\Omega)$, we define $v^{\dagger}(u) = u^{(q)} = [\phi^{\chi}, u] \cdot \chi$.*

1. *For fixed $u \in \mathcal{H}_0^s(\Omega)$, $v^{\dagger}(u)$ is the minimizer of*

$$\begin{cases} \textit{Minimize } \|\psi\| \\ \textit{Subject to } \psi \in \mathcal{H}_0^s(\Omega) \textit{ and } [\phi^{\chi}, \psi] = [\phi^{\chi}, u] \,. \end{cases} \tag{2.2.27}$$

2. *For fixed $u \in \mathcal{H}_0^s(\Omega)$, $v^{\dagger}(u)$ is the minimizer of*

$$\begin{cases} \textit{Minimize } \|u - \psi\| \\ \textit{Subject to } \psi \in \operatorname{span}\{\chi_i : i \in \mathcal{J}\} \,. \end{cases} \tag{2.2.28}$$

3. *For canonical Gaussian field $\xi \sim \mathcal{N}(0, \mathcal{L}^{-1})$,*

$$v^{\dagger}(u) = \mathbb{E}\big[\xi \big| [\phi^{\chi}, \xi] = [\phi^{\chi}, u]\big] \,. \tag{2.2.29}$$

4. *It is true that*[5]

$$v^{\dagger} \in \operatorname{argmin}_{v \in L(\Phi, \mathcal{H}_0^s(\Omega))} \sup_{u \in \mathcal{H}_0^s(\Omega)} \frac{\|u - v(u)\|}{\|u\|} \,. \tag{2.2.30}$$

Pre-Haar wavelet measurement functions

The gamblets used in the subsequent developments will use pre-Haar wavelets (as defined below) as measurement functions $\phi_i^{(k)}$ and our main near-optimal denoising estimates will be derived from their properties (summarized in Thm. 2.2.5).

Let $\delta, h \in (0, 1)$. Let $(\tau_i^{(k)})_{i \in \mathcal{I}^{(k)}}$ be uniformly Lipschitz convex sets forming a nested partition of Ω, i.e., such that $\Omega = \cup_{i \in \mathcal{I}^{(k)}} \tau_i^{(k)}$, $k \in \{1, \ldots, q\}$ is a disjoint union except for the boundaries, and $\tau_i^{(k)} = \cup_{j \in \mathcal{I}^{(k+1)} : j^{(k)} = i} \tau_j^{(k+1)}$, $\quad k \in \{1, \ldots, q-1\}$.

[5] Here $L(\Phi, \mathcal{H}_0^s(\Omega))$ is defined as the set of $\mathcal{H}_0^s(\Omega) \to \mathcal{H}_0^s(\Omega)$ functions that are of form $v(u) = \Psi\big([\phi^{\chi}, u]\big)$ with measureable $\Psi : \mathbb{R}^{\mathcal{J}} \to \mathcal{H}_0^s(\Omega)$.

Assume that each $\tau_i^{(k)}$, contains a ball of radius δh^k, and is contained in the ball of radius $\delta^{-1}h^k$. Writing $|\tau_i^{(k)}|$ for the volume of $\tau_i^{(k)}$, take

$$\phi_i^{(k)} := 1_{\tau_i^{(k)}} |\tau_i^{(k)}|^{-\frac{1}{2}} \,. \tag{2.2.31}$$

The nesting relation (2.2.1) is then satisfied with $\pi_{i,j}^{(k,k+1)} := |\tau_j^{(k+1)}|^{\frac{1}{2}} |\tau_i^{(k)}|^{-\frac{1}{2}}$ for $j^{(k)} = i$ and $\pi_{i,j}^{(k,k+1)} := 0$ otherwise.

For $k \in \{2, \ldots, q\}$, let $\mathcal{J}^{(k)}$ be a finite set of k-tuples of the form $j = (j_1, \ldots, j_k)$ such that $\{j^{(k-1)} \mid j \in \mathcal{J}^{(k)}\} = \mathcal{I}^{(k-1)}$, and for $i \in \mathcal{I}^{(k-1)}$, $\mathrm{Card}\{j \in \mathcal{J}^{(k)} \mid j^{(k-1)} = i\} = \mathrm{Card}\{s \in \mathcal{I}^{(k)} \mid s^{(k-1)} = i\} - 1$. Note that the cardinalities of these sets satisfy (2.2.14).

Write $J^{(k)}$ for the $\mathcal{J}^{(k)} \times \mathcal{J}^{(k)}$ identity matrix. For $k = 2, \ldots, q$, let $W^{(k)}$ be a $\mathcal{J}^{(k)} \times \mathcal{I}^{(k)}$ matrix such that $\mathrm{Im}(W^{(k),T}) = \mathrm{Ker}(\pi^{(k-1,k)})$, $W^{(k)}(W^{(k)})^T = J^{(k)}$ and $W_{i,j}^{(k)} = 0$ for $i^{(k-1)} \neq j^{(k-1)}$.

Theorem 2.2.5. *With pre-Haar wavelet measurement functions, it holds true that*

1. *For $k \in \{1, \ldots, q\}$ and $u \in \mathcal{L}^{-1}L^2(\Omega)$,*

$$\|u - u^{(k)}\| \leq C h^{ks} \|\mathcal{L}u\|_{L^2(\Omega)} \,. \tag{2.2.32}$$

2. *Writing $\mathrm{Cond}(M)$ for the condition number of a matrix M, we have for $k \in \{1, \cdots, q\}$*

$$C^{-1} h^{-2(k-1)s} J^{(k)} \leq B^{(k)} \leq C h^{-2ks} J^{(k)} \tag{2.2.33}$$

and $\mathrm{Cond}(B^{(k)}) \leq C h^{-2s}$.

3. *For $i \in \mathcal{I}^{(k)}$ and $x_i^{(k)} \in \tau_i^{(k)}$,*

$$\|\psi_i\|_{\mathcal{H}^s(\Omega \setminus B(x_i^{(k)}, nh))} \leq C h^{-s} e^{-n/C} \,. \tag{2.2.34}$$

4. *The wavelets $\psi_i^{(k)}, \chi_i^{(k)}$ and stiffness matrices $A^{(k)}, B^{(k)}$ can be computed to precision ϵ (in $\|\cdot\|$-energy norm for elements of $\mathcal{H}_0^s(\Omega)$ and in Frobenius norm for matrices) in $\mathrm{O}(N \log^{3d} \frac{N}{\epsilon})$ complexity.*

Furthermore the constant C depends only on $\delta, \Omega, d, s,$

$$\begin{aligned} \|\mathcal{L}\| &:= \sup_{u \in \mathcal{H}_0^s(\Omega)} \frac{\|\mathcal{L}u\|_{\mathcal{H}^{-s}(\Omega)}}{\|u\|_{\mathcal{H}_0^s(\Omega)}} \text{ and} \\ \|\mathcal{L}^{-1}\| &:= \sup_{u \in \mathcal{H}_0^s(\Omega)} \frac{\|u\|_{\mathcal{H}_0^s(\Omega)}}{\|\mathcal{L}u\|_{\mathcal{H}^{-s}(\Omega)}} \,. \end{aligned} \tag{2.2.35}$$

Proof. (1) and (2) follows from an application of Prop. 4.17 and Theorems 4.14 and 3.19 from [100]. (3) follows from Thm. 2.23 of [100]. 4 follows from the complexity analysis of Alg. 6 of [100]. See [101] for detailed proofs. □

Remark 2.2.6. *The wavelets $\psi_i^{(k)}, \chi_i^{(k)}$ and stiffness matrices $A^{(k)}, B^{(k)}$ can also be computed in $\mathrm{O}(N \log^2 N \log^{2d} \frac{N}{\epsilon})$ complexity using the incomplete Cholesky factorization approach of [119].*

Theorem 2.2.5.2-3 implies that the gamblets are localized both in the eigenspace of operator $\mathcal{L}$ and in Ω space. Further, Theorem 2.2.5.1 shows the accuracy of the recovery, $u^{(k)}$, in $\mathcal{L}$ norm is bounded by L^2 norm of $\mathcal{L}u$. This result is used in the proofs of the denoising result shown in the following section.

2.3 Denoising by truncating the gamblet transform

Near minimax recovery

In this section, we will present the result that truncating the gamblet transform of $\eta = u + \zeta$ in a discrete variant of Problem 5 produces an approximation of u that is minimax optimal up to a multiplicative constant [153, Sec. 4], i.e., *near minimax*. The discretized version of $\mathcal{H}_0^s(\Omega)$ is the finite dimensional space spanned by gamblet wavelets, using pre-Haar measurement functions defined in Sec. 2.2, taken to the q-th level[6]. In addition, the discrete noise used in this problem, $\zeta \in \Psi^{(q)}$, is the projection of the noise (2.1.5) onto $\Psi^{(q)}$ (due to (2.2.8)).

Problem 6. *Let u be an unknown element of $\Psi^{(q)} \subset \mathcal{H}_0^s(\Omega)$ for $q < \infty$. Let ζ be a centered Gaussian vector in $\Psi^{(q)}$ such that*

$$\mathbb{E}\big[[\phi_i^{(q)}, \zeta][\phi_j^{(q)}, \zeta]\big] = \sigma^2 \delta_{i,j} \,. \tag{2.3.1}$$

Given the noisy observation $\eta = u + \zeta$ and a prior bound M on $\|\mathcal{L}u\|_{L^2}$, find an approximation of u in $\Psi^{(q)}$ that is as accurate as possible in the energy norm $\|\cdot\|$.

To justify this discrete approximation, recall that by Theorem 2.2.5, we have $\|u - u^{(q)}\| \le C h^{qs} \|\mathcal{L}u\|_{L^2(\Omega)}$. Hence, with the prior bound on $\|\mathcal{L}u\|_{L^2}$, this approximation is arbitrarily accurate with q large enough. Let η be as in Problem 6 and let gamblets be defined as in Section 2.2 with pre-Haar measurement functions. For $l \in \{1, \ldots, q\}$, let

$$\eta^{(l)} := \sum_{k=1}^{l} [\phi^{(k),\chi}, \eta] \cdot \chi^{(k)} \tag{2.3.2}$$

[6] Note there are no mathematical constraints to the number of levels taken in the decomposition.

and $\eta^{(0)} = 0 \in \Psi^{(q)}$. When $l < q$, $\eta^{(l)}$ is a truncation of the full gamblet transform $\eta = \eta^{(q)} = [\phi^{\chi}, \eta] \cdot \chi$. Let $M > 0$ and write

$$V_M^{(q)} = \{u \in \Psi^{(q)} \mid \|\mathcal{L}u\|_{L^2(\Omega)} \leq M\} \,. \tag{2.3.3}$$

Assume that $\sigma > 0$ and write

$$l^{\dagger} = \text{argmin}_{l \in \{0,\dots,q\}} \beta_l, \tag{2.3.4}$$

for

$$\beta_l = \begin{cases} h^{2s} M^2 & \text{if } l = 0 \\ \sigma^2 h^{-(2s+d)l} + h^{2s(l+1)} M^2 & \text{if } 1 \leq l \leq q-1 \\ h^{-(2s+d)q} \sigma^2 & \text{if } l = q \,. \end{cases} \tag{2.3.5}$$

The following theorem asserts that $\eta^{(l^{\dagger})}$ is a near minimax recovery of u, by which we mean that the $\|\cdot\|^2$ recovery error is minimax optimal up to a multiplicative constant (depending only on $\|\mathcal{L}\|, \|\mathcal{L}^{-1}\|, \Omega, d, \delta$ and whose value can be made explicit using the estimates of [101]). We will also refer to $\eta^{(l^{\dagger})}$ as the smooth recovery of u because, with probability close to 1, it is nearly as regular in energy norm as u.

Theorem 2.3.1. *Suppose $v^{\dagger}(\eta) = \eta^{(l^{\dagger})}$; then there exists a constant C depending only on h, s, $\|\mathcal{L}\|$, $\|\mathcal{L}^{-1}\|$, Ω, d, and δ such that*

$$\sup_{u \in V_M^{(q)}} \mathbb{E}\left[\|u - v^{\dagger}(\eta)\|^2\right] < C \inf_{v(\eta)} \sup_{u \in V_M^{(q)}} \mathbb{E}\left[\|u - v(\eta)\|^2\right], \tag{2.3.6}$$

where the infimum is taken over all measurable functions $v : \Psi^{(q)} \to \Psi^{(q)}$. Furthermore, if $l^{\dagger} \neq 0$, then with probability at least $1 - \varepsilon$,

$$\|\eta^{(l^{\dagger})}\| \leq \|u\| + C\sqrt{\log \frac{1}{\varepsilon}} \sigma^{\frac{2s+d}{4s+d}} M^{\frac{2s+2d}{4s+d}} \,. \tag{2.3.7}$$

Proof. See [153, Sec. 7]. □

Note that $l^{\dagger} = q$ occurs (approximately) when q is such that $h^q > (\frac{\sigma}{M})^{\frac{2}{4s+d}}$, i.e., when

$$\frac{\sigma}{M} < h^{q\frac{4s+d}{2}}, \tag{2.3.8}$$

and in this case $\eta^{(q)}$ is a near minimax optimal recovery of $u^{(q)}$. On the other extreme $l^{\dagger} = 0$ occurs (approximately) when $(\frac{\sigma}{M})^{\frac{2}{4s+d}} > h$, i.e., when

$$\frac{\sigma}{M} > h^{\frac{4s+d}{2}}, \tag{2.3.9}$$

and in this case, the zero signal is a near optimal recovery. The signal-to-noise ratio determines which hierarchical level the truncation occurs. This represents the length-scale of the Gaussian conditioning. This can be seen in

$$\eta^{(k)} = \mathbb{E}\left[\xi \middle| [\phi^{(k)}, \xi] = [\phi^{(k)}, \eta]\right], \tag{2.3.10}$$

which is a conditioning with the level k hierarchical pre-Haar wavelets $\phi^{(k)}$. The trade-off between recovering an overly smooth or noisy signal is illustrated in Fig. 2.2.

Numerical illustrations

Example 2.1.1 with $d = 1$

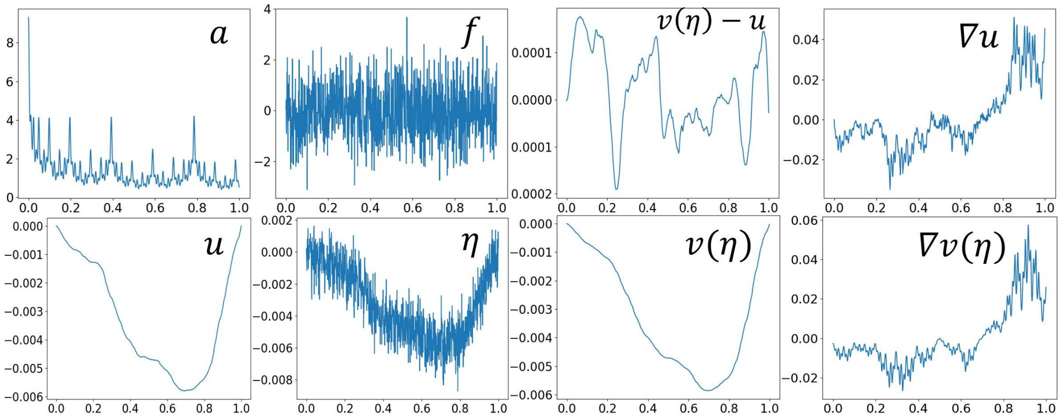

Figure 2.1: [153, Fig. 1], the plots of a, f, u, η, the near minimax recovery $v(\eta) = \eta^{(l^\dagger)}$, its error from u, and the derivatives of u and $v(\eta)$.

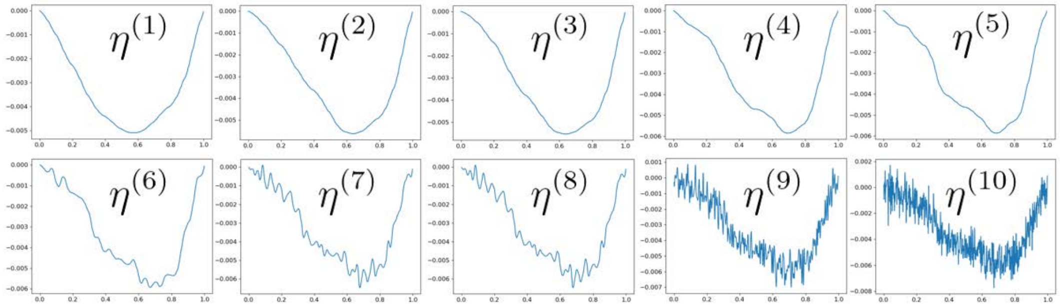

Figure 2.2: A comparison of $\eta^{(l)}$. In this example $l^\dagger = 4$.

Consider Example 2.1.1 with $d = 1$. Take $\Omega = [0, 1] \in \mathbb{R}$, $q = 10$ and $\phi_i^{(k)} = 1_{[\frac{i-1}{2^k}, \frac{i}{2^k}]}$ for $1 \leq i \leq 2^k$. Let $W^{(k)}$ be the 2^{k-1} by 2^k matrix with non-zero entries defined by $W_{i,2i-1} = \frac{1}{\sqrt{2}}$ and $W_{i,2i} = -\frac{1}{\sqrt{2}}$. Let $\mathcal{L} := -\operatorname{div}(a\nabla\cdot)$ with

$$a(x) := \prod_{k=1}^{10}(1 + 0.25\cos(2^k x)). \tag{2.3.11}$$

In Fig. 2.1 we select $f(x)$ at random uniformly over the unit $L^2(\Omega)$-sphere of $\Phi^{(q)}$ and let ζ be white noise (as in (2.1.5)) with $\sigma = 0.001$ and $\eta = u + \zeta$.

We next consider a case where f is smooth, i.e., $f(x) = \frac{\sin(\pi x)}{x}$ on $x \in (0, 1]$ and $f(0) = \pi$. Let ζ be white noise with standard deviation $\sigma = 0.01$. See Fig. 2.3 for the corresponding numerical illustrations.

Both figures show that (1) $v(\eta)$ and $\nabla v(\eta)$ are accurate approximations of u and ∇u (2) the accuracy of these approximations increases with the regularity of f.

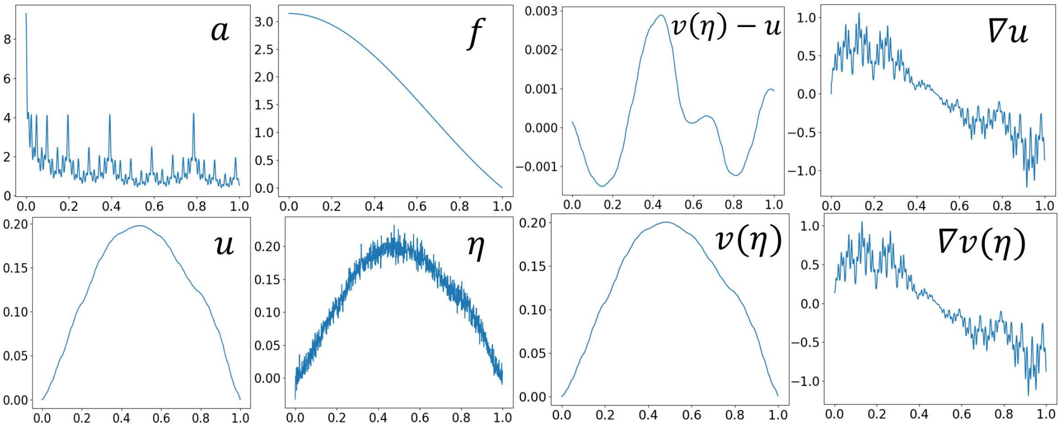

Figure 2.3: [153, Fig. 2], the plots of a, smooth f, u, η, $v(\eta) = \eta^{(l^\dagger)}$, its error from u, and the derivatives of u and $v(\eta)$.

Example 2.1.1 with $d = 2$

Consider Example 2.1.1 with $d = 2$. Take $\Omega = [0, 1]^2$ and $q = 7$. Use the pre-Haar wavelets defined as $\phi_{i,j}^{(k)} = 1_{[\frac{i-1}{2^k}, \frac{i}{2^k}]\times[\frac{j-1}{2^k}, \frac{j}{2^k}]}$ for $1 \le i, j \le 2^k$. Let $W^{(k)}$ be defined be the $3(4^{k-1})$ by 4^k matrix defined as in construction 4.13 of [99].

In Fig. 2.4 we select $f(x)$ at random uniformly over the unit $L^2(\Omega)$-sphere of $\Phi^{(q)}$ and let ζ be white noise (as in (2.1.5)) with $\sigma = 0.001$ and $\eta = u + \zeta$.

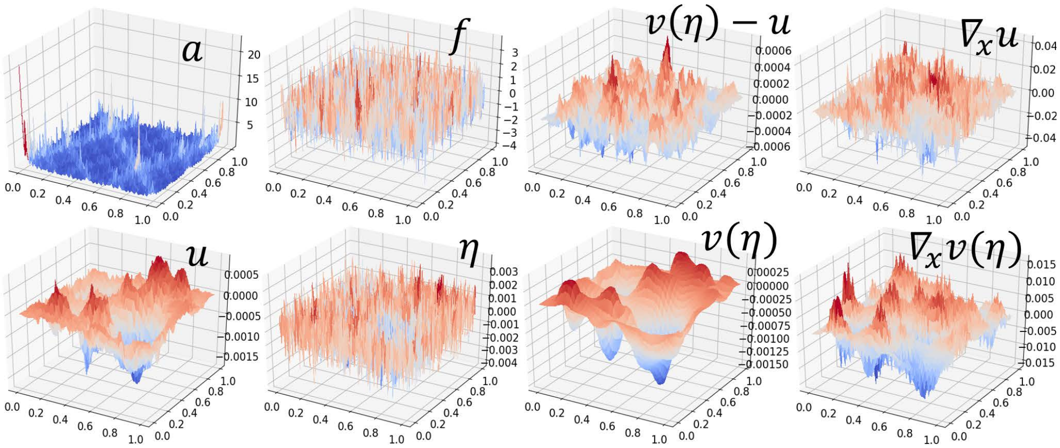

Figure 2.4: [153, Fig. 3], the plots of a, f, u, η, $v(\eta) = \eta^{(l^\dagger)}$, its error from u, and the gradient of u and $v(\eta)$.

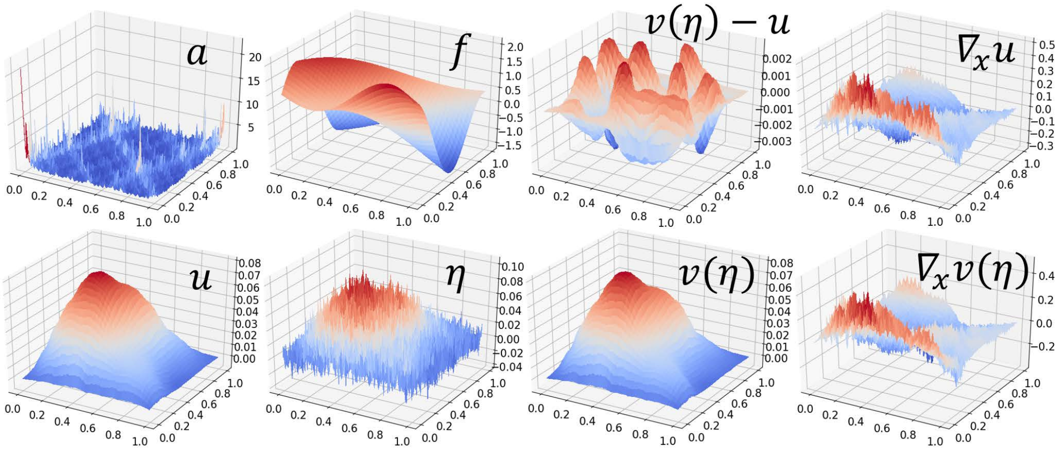

Figure 2.5: The plots of a, smooth f, u, η, $v(\eta) = \eta^{(l^\dagger)}$, its error from u, and the gradient of u and $v(\eta)$ [153, Fig. 4].

Let $\mathcal{L} = -\operatorname{div}(a\nabla\cdot)$ with

$$a(x,y) := \prod_{k=1}^{7} \left[\left(1 + \frac{1}{4}\cos(2^k\pi(x+y)\right)\right. \\ \left.\left(1 + \frac{1}{4}\cos(2^k\pi(x-3y)\right)\right]. \tag{2.3.12}$$

Next consider a case where f is smooth, i.e., $f(x,y) = \cos(3x+y) + \sin(3y) + \sin(7x-5y)$. Let ζ be white noise with standard deviation $\sigma = 0.01$. See Fig. 2.5 for the corresponding numerical illustrations. As with the $d = 1$ plots, the $d = 2$ plots show the accuracy of the recovery of u and ∇u and the positive impact of the regularity of f on that accuracy.

2.4 Comparisons

Hard- and soft-thresholding

Since hard- and soft-thresholding have been used in Donoho and Johnstone [36–38] for the near minimax recovery of regular signals, we will compare the accuracy of (2.3.2) with that of hard- and soft-thresholding the Gamblet transform of the noisy signal [153, Sec. 5]. We call *hard-thresholding* the recovery of u with

$$v(\eta) = \sum_{k=1}^{q} \sum_{i \in \mathcal{J}^{(k)}} H^{t^{(k)}}([\phi_i^{(k),\chi}, \eta])\, \chi_i^{(k)} \tag{2.4.1}$$

and

$$H^{\beta}(x) = \begin{cases} x & |x| > \beta \\ 0 & |x| \leq \beta\,. \end{cases} \tag{2.4.2}$$

We call *soft-thresholding* the recovery of u with

$$v(\eta) = \sum_{k=1}^{q} \sum_{i \in \mathcal{J}^{(k)}} S^{t^{(k)}}([\phi_i^{(k),\chi}, \eta])\, \chi_i^{(k)} \tag{2.4.3}$$

and

$$S^{\beta}(x) = \begin{cases} x - \beta\,\mathrm{sgn}(x) & |x| > \beta \\ 0 & |x| \leq \beta\,. \end{cases} \tag{2.4.4}$$

The parameters $(t_1, \ldots, t_q)$ are adjusted to achieve minimal average errors. Since the mass matrix of ϕ^{χ} is comparable to identity (see [153, Thm. 10]) and the bi-orthogonality identities $[\phi_i^{\chi}, \chi_j] = \delta_{i,j}$, $[f, \chi]$ is approximately uniformly sampled on the unit sphere of $\mathbb{R}^{\mathcal{J}}$ and the variance of $[f, \chi_i^{(k)}]$ can be approximated by $1/|\mathcal{J}|$. Therefore $[\phi^{\chi}, u] = B^{(k),-1}[f, \chi^{(k)}]$ and (2.2.33) imply that the standard deviation of $[\phi^{(k),\chi}, u]$ can be approximated by $h^{-2ks}/\sqrt{|\mathcal{J}|}$. Therefore optimal choices for threshold on the k-th hierarchical level follow the power law $t^{(k)} = h^{-2ks} t_0$ for some parameter t_0.

Regularization

We call *regularization* the recovery of u with $v(\eta)$ defined as the minimizer of

$$\|v(\eta) - \eta\|^2_{L^2(\Omega)} + \alpha \|v(\eta)\|^2\,. \tag{2.4.5}$$

For practical implementation, we consider $A_{i,j} = \langle \tilde{\psi}_i, \tilde{\psi}_j \rangle$, the $N \times N$ stiffness matrix obtained by discretizing $\mathcal{L}$ with finite elements $\tilde{\psi}_1, \ldots, \tilde{\psi}_N$, and write $\eta = \sum_{i=1}^{N} y_i \tilde{\psi}_i$

and $\zeta = \sum_{i=1}^{N} z_i\tilde{\psi}_i$ for the representation of η and ζ over this basis ($\eta = u + \zeta$ and $z \sim \mathcal{N}(0, \sigma^2 I_d)$, writing I_d for the identity matrix). In that discrete setting we have

$$v(\eta) = \sum_{i=1}^{N} x_i\tilde{\psi}_i\,, \tag{2.4.6}$$

where x is the minimizer of

$$|x - y|^2 + \alpha x^T A x\,. \tag{2.4.7}$$

Theorem 2.4.1 and Corollary 2.4.2 show that this recovery corresponds to minimizing the energy norm $\|v\|^2 = x^T A x$, subject to $|x - y| \leq \gamma$ with

$$\gamma = |(I - (\alpha A + I)^{-1})y|\,. \tag{2.4.8}$$

In practice γ would correspond to a level of confidence (e. g., chosen so that $\mathbb{P}[|z| > \gamma] = 0.05$ with $z \sim \mathcal{N}(0, \sigma^2 I_d)$).

Theorem 2.4.1. *Let x be the minimizer of*

$$\begin{cases} \textit{Minimize} & x^T A x \\ \textit{subject to} & |x - y| \leq \gamma\,. \end{cases} \tag{2.4.9}$$

If $|y| \leq \gamma$, then $x = 0$. Otherwise (if $|y| > \gamma$), then $x = (\alpha A + I)^{-1}y$ where α is defined as the solution of (2.4.8).

Proof. Supposing $|y| \leq \gamma$, then if $x = 0$, then $|x - y| \leq \gamma$. Further, $x = 0$ is the global minimum of $x^T A x$. Therefore in this case, $x = 0$.

If $|y| > \gamma$, then at minimum x, the hyperplane tangent to the ellipsoid of center zero must also be tangent to the sphere of center y, which implies that $Ax = \alpha^{-1}(y - x)$ for some parameter α. We therefore have $x = (\alpha A + I)^{-1}y$ and α is determined by the equation $|x - y| = \gamma$, which leads to

$$|(I - (\alpha A + I)^{-1})y| = \gamma\,. \tag{2.4.10}$$

□

Corollary 2.4.2. *If $|y| > \gamma$, then the minimizers of* (2.4.9) *and* (2.4.7) *are identical with α identified as in* (2.4.10).

Proof. $\nabla_x(|x - y|^2 + \alpha x^T A x) = 0$ is equivalent to $x - y + \alpha A x = 0$, which leads to $x = (\alpha A + I)^{-1}y$. □

Numerical experiments

Example 2.1.1 with $d = 1$

Consider the same example as in Subsection 2.3. Table 2.1 shows a comparison of errors measured in L^2 and energy norms averaged over 3,000 independent random realizations of f and ζ (f is uniformly distributed over the unit sphere of $L^2(\Omega)$ and ζ is white noise with $\sigma = 0.001$). The hard variable thresholding recovery is as defined in Section 2.4, regularization recovery is as defined in Section 2.4, and the near minimax recovery refers to $v^\dagger(\eta) = \eta^{(l^\dagger)}$ in Theorem 2.3.1. The best performing algorithm in each category is in bold. In this experiment, the proposed near minimax recovery outperforms the other methods in terms of average error and error variance.

Algorithm	$\mathcal{L}$ Error AVG	$\mathcal{L}$ Error STDEV	L^2 Error AVG	L^2 Error STDEV
Hard variable threshold	4.78×10^{-3}	9.64×10^{-4}	2.25×10^{-4}	1.07×10^{-4}
Soft variable threshold	4.27×10^{-3}	7.70×10^{-4}	1.65×10^{-4}	5.63×10^{-5}
Regularization recovery	4.37×10^{-3}	7.93×10^{-4}	2.82×10^{-4}	7.83×10^{-5}
Near minimax recovery	$\mathbf{3.90 \times 10^{-3}}$	$\mathbf{5.30 \times 10^{-4}}$	$\mathbf{1.24 \times 10^{-4}}$	$\mathbf{2.50 \times 10^{-5}}$

Table 2.1: Comparison of the performance of denoising algorithms for $d = 1$.

For reference, the average and standard deviation of the (discrete) energy norm of ζ used in this trial were 1.68 and 0.06, respectively.

Example 2.1.1 with $d = 2$

Consider the same example as in Subsection 2.3. Table 2.2 shows errors measured in L^2 and energy norms averaged over 100 independent random realizations of f and ζ (f is uniformly distributed over the unit sphere of $L^2(\Omega)$ and ζ is white noise with $\sigma = 0.001$). In this experiment, the proposed near minimax recovery is the best or near the best in every error metric (it is slightly outperformed by regularization in average $\mathcal{L}$ error).

For reference, the average and standard deviation of the (discrete) $\mathcal{L}$ norm of this trial's ζ were 0.250 and 0.06, respectively.

Algorithm	$\mathcal{L}$ Error AVG	$\mathcal{L}$ Error STDEV	L^2 Error AVG	L^2 Error STDEV
Hard variable threshold	6.95×10^{-3}	9.78×10^{-5}	1.42×10^{-4}	7.76×10^{-6}
Soft variable threshold	7.18×10^{-3}	1.57×10^{-4}	1.90×10^{-4}	2.35×10^{-5}
Regularization recovery	$\mathbf{6.90\times10^{-3}}$	1.03×10^{-4}	1.86×10^{-4}	1.88×10^{-5}
Near minimax recovery	6.94×10^{-3}	$\mathbf{9.58\times10^{-5}}$	$\mathbf{1.40\times10^{-4}}$	$\mathbf{7.29\times10^{-6}}$

Table 2.2: Comparison of the performance of denoising algorithms for $d = 2$.

Chapter 3

THE MODE DECOMPOSITION PROBLEM

This chapter will be devoted to presenting Empirical Mode Decomposition (EMD) and Synchrosqueezing transform (SST) algorithms. To introduce these topics, we give the following prototypical mode decomposition problem, with an example illustrated in Fig. 3.1.

Problem 7. *For $m \in \mathbb{N}^*$, let $a_1, \ldots, a_m$ be piecewise smooth functions on interval $I \subset \mathbb{R}$ and let $\theta_1, \ldots, \theta_m$ be strictly increasing functions on I. Assume that m and the a_i, θ_i are unknown. Given the observation of $v(t) = \sum_{i=1}^m a_i(t)\cos\big(\theta_i(t)\big), t \in I$, recover the modes $v_i(t) := a_i(t)\cos\big(\theta_i(t)\big)$.*

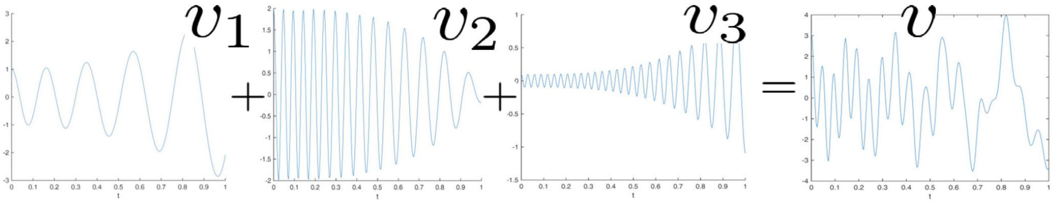

Figure 3.1: [102, Fig. 1], a prototypical mode decomposition problem: given $v = v_1 + v_2 + v_3$ recover v_1, v_2, v_3.

In practical applications, the *instantaneous amplitudes* and *frequencies*, i.e., a_i and $\omega_i = \frac{d\theta_i}{dt}$, are generally assumed to be smooth and well separated. Furthermore, a_i and ω_i are usually assumed to be varying at a slower rate than the *instantaneous phases* θ_i, i.e., $|\frac{da_i}{dt}| \leq \epsilon|\frac{d\theta_i}{dt}|$ and $|\frac{d\omega_i}{dt}| \leq \epsilon|\frac{d\theta_i}{dt}|$.

The analysis of this family of signals is found in a wide variety of scientific fields, a broad exposition of which can be found in [67]. We will briefly summarize applications in the natural sciences, beginning with meteorology. For instance, time signals stemming from the geopotential height[1] can be analyzed. This signal can be decomposed into modes with differing frequencies [22] and [67, Sec. 10]. The separated modes are found to correspond to effects from yearly seasonal variability, the Quasi-Biennial Oscillation (QBO), the El-Niño–Southern Oscillation (ENSO),

[1] i.e., the altitude corresponding to a certain air pressure in Earth's lower atmosphere.

and the solar cycle. The approximate periods of these oscillations are 1, 2, 4, and 11 years, respectively. These long term climatic effects can also be extracted from temperature and precipitation data as in [70, 84, 158] and [67, Sec. 12]. The signal corresponding to local sea-level data can also be separated into short-term effects (such as tides, storm surge, seasonal temperature and precipitation); long-term, multiyear oscillations (as mentioned in the previous example); and global sea-level rise [74, 141] and [67, Sec. 9]. Furthermore, the recovered modes of local weather statistics, such as temperature, humidity, pressure, etc., were found to have associations with the incidence of headaches in [150, 151].

Another area where EMD is of importance is in the study of seismological signals [58, 63, 147]. Specific applications include the denoising of such signals [50, 55] and the identification of geological features, such as faults, sand boundaries, or resources, by using seismic reflection data [9, 68, 148]. EMD also can use acceleration readings in buildings to assess structural damage in seismic events, as discussed in [146, 152] and [67, Sec. 14]. Moreover, the structural integrity of bridges can be analyzed by applying EMD to the response from a passing vehicle [95] and [67, Sec. 15]. Mode decomposition techniques are also applicable in astronomical signals. Examples of such include the study of solar atmosphere oscillation [78], X-ray binary systems [35], and satellite orbital drift [67, Sec. 11]. Nearly periodic signals also occur in oceanography including in the classification of marine mammal vocal signals [121], the ocean's electromagnetic fields [15], and the analysis of ocean waves [67, Sec. 13]. Further, EMD can be used to help process images of ocean waves [67, Sec. 16].

Finally, EMD is a useful tool with medical data, including ECG and EEG signals, i.e., heart and brain electrical activity, as well as epidemiologic statistics. ECG signals can be analyzed with EMD to distinguish healthy patients from those with cardiac arrhythmia [117]. Such ECG signals can also be denoised to remove noise and other measurement artifacts [14]. Analogously, EEG signals can be decomposed into modes [93] to help distinguish healthy and epileptic signals [34]. Such analysis can also aid in the development of a brain-computer interface [32], which maps EEG signals to physical movements, or an emotion-recognition algorithm [91]. In addition, emotional recognition can also be accomplished via the analysis of vocal waves [62]. Moreover, EMD can be used to analyze epidemiological data such as the spatial-temporal dynamics of the incidence of dengue hemorrhagic fever and provides information on the processes that contribute to its spread [23]. The remainder of this chapter will be devoted to discussing EMD, SST, and their variants.

3.1 Hilbert-Huang transform

Empirical mode decomposition

Algorithm 1 Empirical Mode Decomposition

```
**Input:** $v$
$i, j \leftarrow 1, 1$
$v_{i,j} \leftarrow v$
**while** STOP_OUTER == FALSE **do**
    **while** STOP_INNER == FALSE **do**
        Identify all local maxima and minima of $v_{i,j}$; then fit both sets of points to
        a cubic spline and denote as $u(t)$ and $l(t)$, respectively.
        $m \leftarrow \big(u + l\big)/2$
        $v_{i,j+1} \leftarrow v_{i,j} - m$
        $j \leftarrow j + 1$
        **if** $v_{i,j}$ is an IMF (or similar stopping condition) **then**
            STOP_INNER ← TRUE
            $v_i \leftarrow v_{i,j}$
            $i, j \leftarrow i + 1, 1$
            $v_{i,j} \leftarrow v - v_1 - \cdots - v_{i-1}$
        **end if**
    **end while**
    **if** $v_{i,j}$ is small or monotonic (or similar stopping condition) **then**
        STOP_OUTER ← TRUE
        $n \leftarrow i$
        $r \leftarrow v - v_1 - \cdots - v_n$
    **end if**
**end while**
**Output:** $v_1, \ldots, v_n, r$
```

The Hilbert-Huang Transform (HHT) consists of both EMD and an application of the Hilbert transform. While it is much more widely used in applications than is synchrosqueezing, it is known that theoretical analysis of the results is difficult. The input of EMD is a signal defined over some interval $I \subset \mathbb{R}$, i.e., $v : I \to \mathbb{R}$, which is assumed to be of the form given in Problem 7. It outputs a decomposition of v,

$$v(t) = v_1(t) + v_2(t) + \cdots + v_n(t) + r(t)\,, \tag{3.1.1}$$

where $v_1, \ldots, v_n$ are *intrinsic mode functions* (IMF), which are defined to be such that (1) over I, the number of extrema and the number of zero-crossings differ by at most one (2) at any point, the mean value of the envelope defined by the local maxima and the envelope defined by the local minima is zero [67, Sec. 1]. Next, Hilbert spectral analysis (HSA) is applied to these IMF's to express them in the form $a(t)\cos(\theta(t))$.

The methodology of EMD, as outlined in Algorithm 1, will be discussed next. Begin by inputting signal v into the algorithm as in step 1. The algorithm will use i as the index of the outer loop (steps 4 to 22), each step of which computes one IMF. Meanwhile j will be used to index the inner loop (steps 5 to 16), which refines the estimate of each IMF until it satisfies some stopping condition. This refinement process aims to remove lower frequency modes to isolate the highest frequency and is referred to as *sifting*. Each $v_{i,j}$ can be interpreted as residuals of the signal, which is initialized as the original signal $v_{1,1} = v$ in step 3. The inner loop consists of first identifying all local maxima and minima of residual signal $v_{i,j}$ in step 6. Then, this set of maxima and minima points will be used to interpolate $u(t)$ and $l(t)$, respectively, with cubic splines, which are interpretable as the upper and lower envelope of $v_{i,j}$. The mean of these envelopes is $m = (u + l)/2$ as in step 7. An example of $v_{i,j}, u, l, m$ is plotted in Figure 3.2. In steps 8 and 9, this estimate of the mean is removed from $v_{i,j}$ to create the next residual signal $v_{i,j+1}$ and the inner index j is updated. It is then checked whether the new residual signal satisfies a stopping condition in step 11, and if so the residual signal is added as an identified

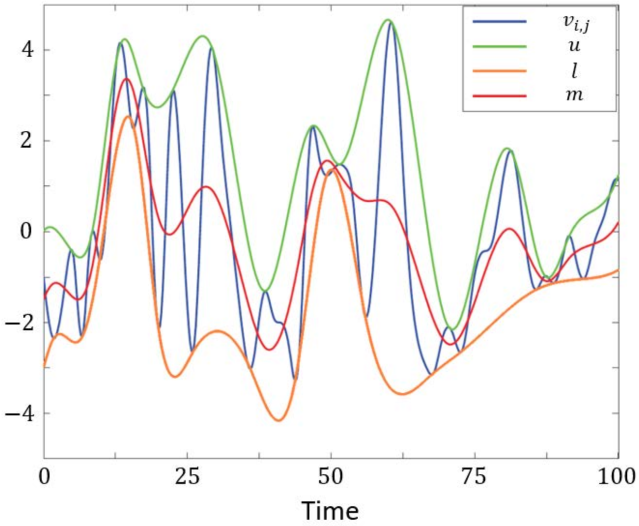

Figure 3.2: Upper and lower envelopes of residual signal $v_{i,j}$ are u and l. The mean of the envelopes is m. Figure adapted from [67, Fig. 1.2] with permission.

IMF, v_i. Finally, it is checked whether residual signal $v_{i,1} = v - v_1 - \cdots - v_{i-1}$ is small or monotonic at step 17, and if so, the algorithm is terminated. The extracted IMF's $v_1, \dots v_n$ and residual $r = v - v_1 - \cdots - v_n$ are returned. These modes can be converted into form $a(t)\cos(\theta(t))$ with the Hilbert transform, which will be presented in the next section.

There is little theoretical backing for the sifting process. The convergence of sifting has not been established and is left as a conjecture in [76, Hyp. 2,3]. With the assumption of convergence, however, it is known that the highest frequency mode remaining is sifted [76]. A major source of difficulty for analysis stems from the use of cubic splines [27, 51]. Various approaches have been used to avoid this difficulty, including the derivation of such convergence bounds when using trigonometric interpolation [60]. Furthermore, in practice it is known that too many siftings can overly smooth uneven amplitudes, leading to the need for more sophisticated stopping conditions, such as Cauchy-type convergence tests or the number of consecutive times the numbers of zero-crossings and extrema are unchanged [66, pg. 920] [67, pg. 8-9].

Hilbert transform

Following the presentation in [67, Sec. 1.2], the Hilbert transform estimates the complex component of any real mode $x(t)$. Specifically, if we define

$$y(t) := \mathcal{H}[x(t)] = \frac{1}{\pi}\mathrm{PV}\int_{-\infty}^{\infty} \frac{x(\tau)}{t-\tau} d\tau \tag{3.1.2}$$

as the Hilbert transform of x, then the analytic signal defined as

$$z(t) := x(t) + iy(t) = a(t)e^{i\theta(t)}, \tag{3.1.3}$$

where

$$a(t) = \sqrt{x^2 + y^2}, \text{ and } \theta(t) = \arctan\left(\frac{y}{x}\right). \tag{3.1.4}$$

Notice that this implies that $x(t) = a(t)\cos(\theta(t))$ where $a(t)$ and $\theta(t)$ are called the instantaneous amplitude and phase, respectively. Furthermore, the instantaneous frequencies can be derived as $\omega(t) = \frac{d\theta}{dt}$. This illustrates how derived IMF's can be converted into form $a(t)\cos(\theta(t))$ and local properties of the oscillation can be estimated.

3.2 Synchrosqueezing transform

An algorithmic description of the *Synchrosqueezing Transform* (SST) will be given in this section. This method aims to address the mode decomposition problem with a more mathematical framework than EMD [24]. In this setting, intrinsic mode functions are defined to be of the form $a(t)e^{i\theta(t)}$, where a and θ satisfy smoothness conditions as given in [24, Thm. 3.1].

Accuracy bounds are proven for functions in class $\mathcal{A}_{\epsilon,d}$, which are superpositions of intrinsic mode functions

$$v(t) = \sum_{k=1}^{K} v_k(t) = \sum_{k=1}^{K} a_k(t)e^{i\theta_k(t)}\,, \tag{3.2.1}$$

with conditions on the separation of instantaneous frequencies $\theta_k'(t)$. The parameter ϵ bounds the rate of change of amplitude and frequency and d the separation between the instantaneous frequencies of modes. In applications, synchrosqueezing can be applied to signals not in this class, such as signals corrupted by noise, though theoretical accuracy bounds would not apply. The methodology utilizes frequency-reallocation methods to estimate instantaneous frequencies of modes. The estimates are then used in a wavelet-reconstruction formula to obtain estimates of each mode.

Algorithm 2 Synchrosqueezing

```
1: **Input:** $v$
2: **Algorithm parameters:** Wavelet $\psi$ with compactly supported Fourier transform and bump function $h$ with unit integral.
3: $W_v^\psi(a,b) \leftarrow \int v(t)a^{-1/2}\psi^*\left(\frac{t-b}{a}\right)dt$
4: $\omega_v(a,b) \leftarrow -i(W_v^\psi(a,b))^{-1}\frac{\partial}{\partial b}W_v^\psi(a,b)$
5: $A_{v,\epsilon}(b) \leftarrow \{a \in \mathbb{R}^+ : |W_v^\psi(a,b)| > \epsilon\}$
6: $S_{v,\epsilon}^\delta(\omega,b) \leftarrow \int_{A_{v,\epsilon}(b)} W_v^\psi(a,b)\frac{1}{\delta}h\left(\frac{\omega-\omega_v(a,b)}{\delta}\right)a^{-3/2}da$
7: Divide time-frequency domain into $K$ bands, each corresponding to a neighborhood of instantaneous frequencies of each mode by observing $S_{v,\epsilon}^\delta(\omega,b)$. Denote bands as $B_1,\ldots,B_k$.
8: $\mathcal{R}_\psi = \sqrt{2\pi}\int \hat{\psi}(\zeta)\zeta^{-1}d\zeta$
9: $v_{k,e}(t) \leftarrow \mathcal{R}_\psi^{-1}\lim_{\delta\to 0}\left(\int_{(\omega,t)\in B_k} S_{v,\epsilon}^\delta(\omega,t)d\omega\right)$
10: **Output:** $v_{1,e},\ldots,v_{K,e}$
```

Algorithm 2 outlines the methodology in SST in the continuous setting. In applications, discrete approximations of the expressions are used [147, Sec. 2.2]. The input of the SST algorithm is a signal v as shown in step 1. The algorithm utilizes a mother wavelet ψ in the Schwartz class, with Fourier transform supported on $[1-\Delta, 1+\Delta]$ and a unit integral bump function $h \in C_c^\infty$. In step 3, the *continuous wavelet transform* (CWT) of signal v with frequency scale a and time b is computed as

$$W_v^{\psi}(a,b) = \int v(t)a^{-1/2}\psi^*\Big(\frac{t-b}{a}\Big)dt\,, \tag{3.2.2}$$

where ψ^* is the complex conjugate of wavelet ψ. As can be observed in Figure 3.3.2, this transform has relatively large norm when frequency scale, a, approximately aligns with the instantaneous frequency of a mode, i.e., θ_k'. This transform has the drawback of not sharply estimating the instantaneous frequencies, which is addressed by synchrosqueezing. Step 4 calculates the instantaneous frequency of the CWT at time-frequency position (a,b). This is then used to reallocate CWT frequencies by mapping $(a,b) \to (\omega_v(a,b),b)$. Steps 5 and 6 show this reallocation, where all time-frequency points with CWT norms above cut-off ϵ are redirected to synchrosqueezed CWT $S_{v,\epsilon}^{\delta}(\omega,b)$. As can be seen in Figure 3.3.3, this leads to a more concentrated image of instantaneous frequencies. With this image, in step 7, the time-frequency plane is split into bands corresponding to each mode. Although this can require human judgement, the algorithm is not sensitive to the choice of band so long as one band contains an entire single mode and does not contain multiple modes. The bands and the synchrosqueezed CWT are then used to reconstruct each signal in steps 8 and 9. These reconstruction formulas are inspired from applying Fourier analysis in the case with pure tone $v = A\cos(\omega t)$ [24, Sec. 2]. Bounds on the recovery errors are proven in the setting where $v \in \mathcal{A}_{\epsilon,d}$, i.e., a composition of modes with well separated frequencies, in [24, Sec. 3]. Finally, in practice, one must shift to discrete analogues of these equations, which is outlined in [147, Sec. 2.2].

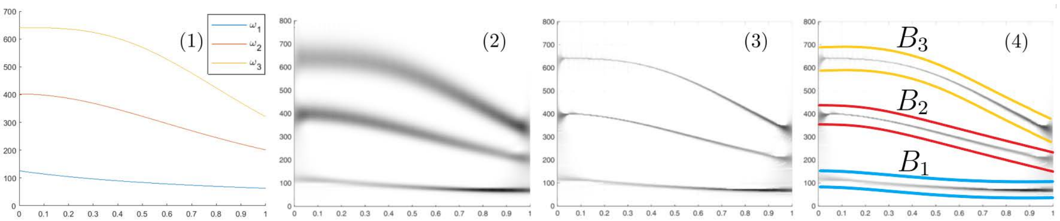

Figure 3.3: Signal v is the composition of 3 modes where the following is plotted in time-frequency domain: (1) the instantaneous frequencies of each mode (2) the norm of the continuous wavelet transform (CWT) of v, $|W_v^{\psi}(a,b)|$ (3) the synchrosqueezed CWT (4) the bands corresponding to each mode.

3.3 Extensions and further approaches

One of the major issues of EMD in practice is *mode mixing*, which refers to either when one IMF contains modes with differing frequencies or when a mode is split between multiple IMFs. Mode mixing is prominent when modes are intermittent, i.e., supported over a subset of the time domain. An extension of the algorithm has been developed to address this issue called *Ensemble Empirical Mode Decomposition* (EEMD) [142]. The main idea is to construct an ensemble of IMFs by applying EMD to the signal corrupted with random realizations of white noise. The ensemble is then averaged to obtain estimates of the modes.

Another issue with EMD is its lack of robustness to signal noise. This is addressed by thresholding EEMD modes [49]. Further, replacing the sifting process with an optimization approach has stronger theoretical backing and convergence guarantees [65, 90, 108]. This approach has been found to improve robustness to noise and sampling[2] effects [112]. More recently, *Variational Mode Decomposition* applies an optimization of mode bandwidth[3] to concurrently estimate modes which also shows robustness to noise and sampling effects in practical examples [40].

Further developments to the SST that are outlined in [89] will be discussed next. The SST described in Section 3.2 is commonly known as the CWT SST or WSST. A variant of this method is the *Short-time Fourier Transform* (STFT) SST, which is typically referred to as FSST [134]. This replaces the CWT calculation in (3.2.2) with the STFT on signal f with window $g \in L^{\infty}(\mathbb{R})$:

$$V_f^g(t,\omega) = \int f(\tau)g(\tau - t)e^{-2\pi i\omega(\tau - t)}d\tau \,. \tag{3.3.1}$$

It is notable that the window, g, has fixed width relative to frequency ω, unlike in CWT where the width is inversely proportional to frequency. This leads to the main difference between the methods where FSST and WSST have absolute and relative frequency resolution of modes, respectively [89, Sec. 3.3]. Improvements in robustness to noise are made in *ConceFT* [25] with use of multitapering. This can be defined in both the WSST and FSST contexts. In the WSST context, multiple wavelets are selected, the SST of each wavelet is calculated, and the results are averaged. ConceFT can be defined analogously using FSST. Another difficulty associated with SST is robustness to frequency modulation, i.e., when the rate of change of instantaneous frequency of a mode, $|\theta_k''(t)|$, is large. This is addressed

[2] Effects stemming from the fact that signals are not continuously measured in practical applications.

[3] Loosely, the variation of frequency and amplitude.

using the *second-order synchrosqueezing transform* [11, 94]. In this extension of SST, $W_\nu^{t\psi}$ is used in conjunction with W_ν^{ψ} to generate a reassignment of both time and frequency in the synchrosqueezing of the CWT (or analogously for the STFT). This can be taken further in the *higher-order synchrosqueezing transform* [107] where $W_\nu^{t^k\psi}$ or $V_\nu^{t^k g}$ are used to obtain n-th order reallocation estimates.

Chapter 4

ITERATED MICRO-LOCAL KERNEL MODE DECOMPOSITION FOR KNOWN BASE WAVEFORMS

This chapter will introduce iterated micro-local kernel mode decomposition (KMD) [102, Sec. 8], which is a GP-inspired approach to the mode decomposition problem, i.e., Prob. 7. We present its adaptability to address generalizations such as possibly unknown, non-trigonometric waveforms and modes with crossing frequencies or vanishing amplitudes. The method borrows the sequential peeling of modes from EMD and uses a variant of the SST to identify mode frequencies. We will present the methodology behind mode identification and estimation by introducing the algorithm in the context of mode recovery with known waveforms as in Problem 8.

Problem 8. *For $m \in \mathbb{N}^*$, let $a_1, \ldots, a_m$ be piecewise smooth functions on $[-1, 1]$, let $\theta_1, \ldots, \theta_m$ be strictly increasing functions on $[-1, 1]$, and let y be a square-integrable 2π-periodic function. Assume that m and the a_i, θ_i are unknown and the base waveform y is known. We further assume that, for some $\epsilon > 0$, $a_i(t) > \epsilon$ and that $\dot{\theta}_i(t)/\dot{\theta}_j(t) \notin [1-\epsilon, 1+\epsilon]$ for all i, j, t. Given the observation $v(t) = \sum_{i=1}^m a_i(t) y\big(\theta_i(t)\big)$ (for $t \in [-1, 1]$) recover the modes $v_i := a_i(t) y\big(\theta_i(t)\big)$.*

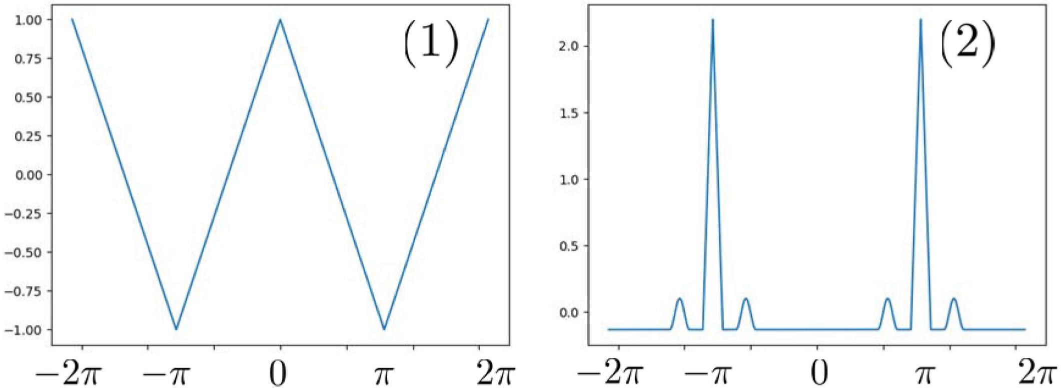

Figure 4.1: [102, Fig. 23], (1) triangle base waveform (2) EKG base waveform.

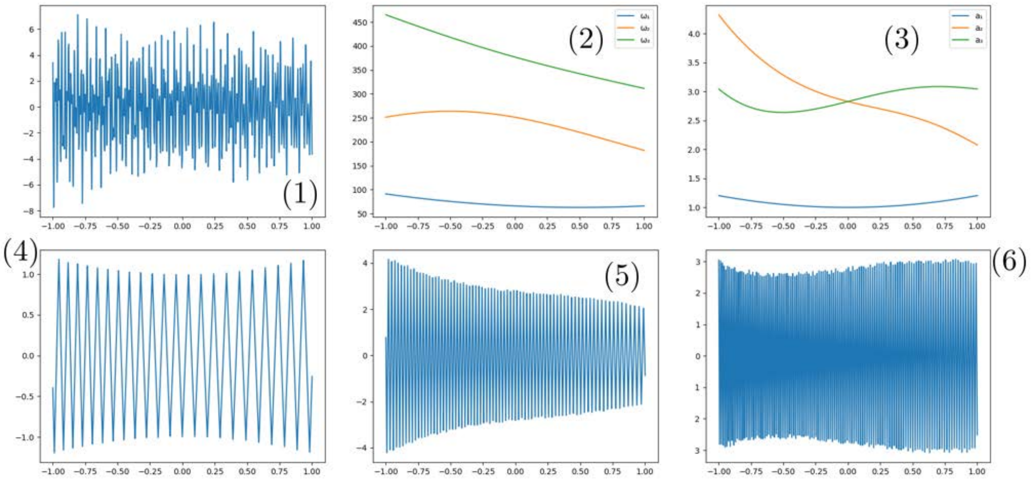

Figure 4.2: [102, Fig. 24], triangle base waveform: (1) Signal v (2) Instantaneous frequencies $\omega_i := \dot{\theta}_i$ (3) Amplitudes a_i (4, 5, 6) Modes v_1, v_2, v_3.

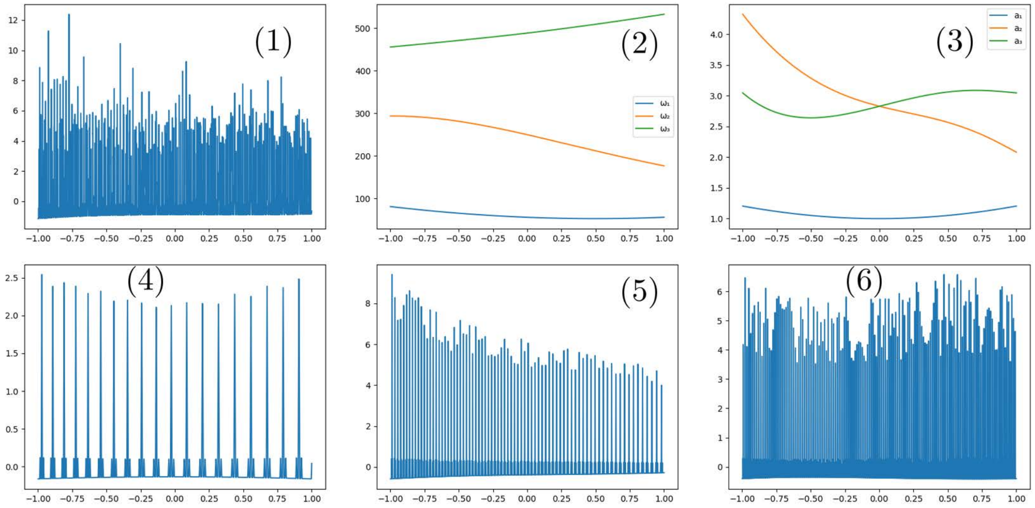

Figure 4.3: [102, Fig. 25], EKG base waveform: (1) Signal v (2) Instantaneous frequencies $\omega_i := \dot{\theta}_i$ (3) Amplitudes a_i (4, 5, 6) Modes v_1, v_2, v_3.

Example 4.0.1. *Figure 4.1 shows two full periods of two 2π-periodic base waveforms (triangle and EKG), which we will use in our numerical experiments/illustrations. The EKG (-like) waveform is $\big(y_{EKG}(t) - (2\pi)^{-1}\int_0^{2\pi} y_{EKG}(s)\,ds\big)/\|y_{EKG}\|_{L^2([0,2\pi))}$ with $y_{EKG}(t)$ defined on $[0,2\pi)$ as (1) $0.3 - |t-\pi|$ for $|t-\pi| < 0.3$ (2) $0.03\cos^2(\frac{\pi}{0.6}(t-\pi+1))$ for $|t-\pi+1| < 0.3$ (3) $0.03\cos^2(\frac{\pi}{0.6}(t-\pi-1))$ for $|t-\pi-1| < 0.3$ and (4) 0 otherwise.*

To aid in the exposition, we present the algorithm as a network assembled with elementary modules. Our approach is summarized in Algorithm 3 and explained in the following sections. The algorithm iterates modules described in (1) to (3) to estimate each mode, $v_i \approx v_{i,e}$. We denote $v^{(i)} = v - v_{1,e} - \cdots - v_{i-1,e}$ as signal v after the first $i - 1$ mode estimates are peeled. Beginning with $i = 0$ and $v^{(0)} = v$, the algorithm proceeds as follows:

1. Use the max-pool energy $\mathcal{S}$ (4.1.6) to obtain an estimate of the instantaneous phase and frequency, $\theta_{\text{low}}(v^{(i)})$ and $\omega_{\text{low}}(v^{(i)})$ associated with the lowest instantaneous frequency (as described in Section 4.1).

2. Iterate a *micro-local* KMD (presented in Section 4.2) of the signal $v^{(i)}$ to obtain a highly accurate estimate of the phase/amplitude θ_k, a_k of their corresponding mode v_k for all $k \leq i$ (this iteration can achieve near machine-precision accuracies when the instantaneous frequencies are separated).

3. Peel off the mode v_i from $v^{(i)}$, i.e., $v^{(i+1)} = v^{(i)} - v_i$ and update index $i \to i+1$.

4. Iterate 1-3 to obtain all the modes.

5. Perform a last micro-local KMD of the signal for higher accuracy.

To illustrate this approach we will apply it to the signals v displayed in Figures 4.2 and 4.3, where the modes of Figure 4.2 are triangular and those of Figure 4.3 are EKG.

4.1 Max-pooling and the lowest instantaneous frequency

We will now present a variant of the SST [24] used for identifying mode frequencies. It will be applied in a module identifying the phase and instantaneous frequency of the lowest frequency mode [102, Sec. 8.2]. Begin by defining wavelets

$$\begin{aligned}\chi_{\tau,\omega,c}(t) &:= \left(\frac{2}{\pi}\right)^{\frac{1}{4}}\sqrt{\frac{\omega}{\alpha}}\cos(\omega(t-\tau))e^{-\frac{\omega^2(t-\tau)^2}{\alpha^2}}, \quad t \in \mathbb{R},\\ \chi_{\tau,\omega,s}(t) &:= \left(\frac{2}{\pi}\right)^{\frac{1}{4}}\sqrt{\frac{\omega}{\alpha}}\sin(\omega(t-\tau))e^{-\frac{\omega^2(t-\tau)^2}{\alpha^2}}, \quad t \in \mathbb{R}, \qquad (4.1.1)\end{aligned}$$

as well as complex wavelet $\chi_{\tau,\omega}(t) = \chi_{\tau,\omega,c}(t) - i\chi_{\tau,\omega,s}(t)$. Note that τ indicates the time of the center of the wavelet within $[-1, 1]$ while ω represents the frequency. Then the continuous wavelet transform (CWT) as in [24] of signal f at (τ, ω) is defined as

$$W(\tau, \omega, f) := \int_{-1}^{1} \chi_{\tau,\omega}(t) f(t) dt\,. \qquad (4.1.2)$$

We further define cosine and sine analogues of the CWT as

$$\begin{aligned} W_c(\tau,\omega,f) &:= \int_0^1 \chi_{\tau,\omega,c}(t) f(t)\, dt \\ W_s(\tau,\omega,f) &:= \int_0^1 \chi_{\tau,\omega,s}(t) f(t)\, dt\,. \end{aligned} \tag{4.1.3}$$

The energy of the signal in (τ,ω)-space is then defined by

$$E(\tau,\omega,f) := |W(\tau,\omega,f)|^2\,. \tag{4.1.4}$$

Mimicking the instantaneous phase and frequency estimation in SST, we define

$$\begin{aligned} \theta_e(\tau,\omega,f) &:= \mathrm{phase}(W(\tau,\omega,f)) \\ \omega_e(\tau,\omega,f) &:= \frac{\partial\theta_e}{\partial\tau}(\tau,\omega,f)\,. \end{aligned} \tag{4.1.5}$$

We introduce the max-pool energy

$$\mathcal{S}(\tau,\omega,f) = \max_{\omega':\omega_e(\tau,\omega')=\omega} E(\tau,\omega',f) \tag{4.1.6}$$

as a variant of the SST which avoids the dependence on the choice of measure, as in the remark of [24, Eq. 2.7]. This max-squeezing can also be interpreted in an additive-kernel setting with further details, illustrations, and comparisons to SST in [102, Sec. 4].

This calculation of signal energy in (τ,ω)-space is then used to design modules which take the signal as input and return an estimate of the instantaneous phase and frequency of the lowest-frequency mode, $\theta_{\mathrm{low}}(f)$ and $\omega_{\mathrm{low}}(f)$. Note that both of the outputs are $[-1,1]\to\mathbb{R}$ functions. We restrict our presentation to the situation where the instantaneous frequencies $\dot\theta_i$ do not cross each other. The main steps of the computation performed by this module are as follows. Let $\mathcal{S}(\tau,\omega,f)$ be the max-pool energy defined as in (4.1.6). Then, let A_{low} be defined to be a subset of the time-frequency domain (τ,ω) identified (as in Figure 4.4.2) as a narrow sausage band around the lowest instantaneous frequency defined by the local maxima of the $\mathcal{S}(\tau,\omega,f)$. If no modes can be detected (above a given threshold) in $\mathcal{S}(\tau,\omega,f)$ then we set $\theta_{\mathrm{low}}(f)=\emptyset$. Otherwise, we let

$$\omega_{\mathrm{low}}(f)(\tau) := \omega_e\big(\tau, \mathrm{argmax}_{\omega:(\tau,\omega)\in A_{\mathrm{low}}} \mathcal{S}(\tau,\omega,f)\big) \tag{4.1.7}$$

be the estimated instantaneous frequency of the mode having the lowest instantaneous frequency and, with θ_e defined as in (4.1.5), let

$$\theta_{\mathrm{low}}(f)(\tau) := \theta_e(\tau,\omega_{\mathrm{low}}(f)(\tau),f) \tag{4.1.8}$$

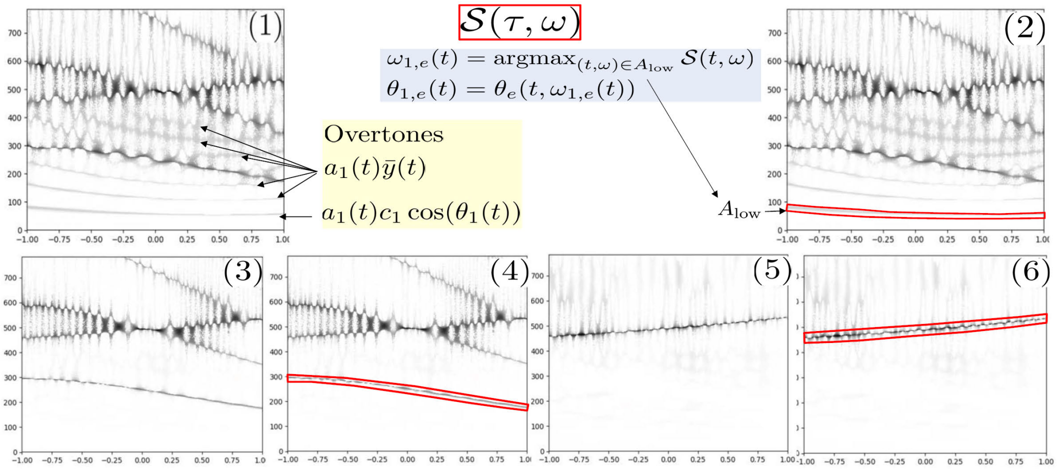

Figure 4.4: [102, Fig. 26], max-squeezing with the EKG base waveform and derivation of the instantaneous phase estimates $\theta_{i,e}$. (1,2) $(\tau,\omega)\to\mathcal{S}(\tau,\omega,v)$ and identification of A_{low} (3, 4) $(\tau,\omega)\to\mathcal{S}(\tau,\omega,v-v_{1,e})$ and identification of its A_{low} (5,6) $(\tau,\omega)\to\mathcal{S}(\tau,\omega,v-v_{1,e}-v_{2,e})$ and identification of its A_{low}.

be the corresponding estimated instantaneous phase. Notationally, we sometimes leave out f and write ω_{low} or θ_{low} when unambiguous.

4.2 The micro-local KMD module

We will now present the micro-local KMD module [102, Sec. 8.1], which will estimate amplitudes and refine SST phase estimates. As input, it takes a time τ, an estimated phase function of a mode θ_e, and signal f. Suppose the lowest frequency mode of f is of form $v_{\mathrm{low}}(t)=a_{\mathrm{low}}(t)y(\theta_{\mathrm{low}}(t))$, and its phase is estimated as $\theta_{\mathrm{low},e}$. The module outputs an estimate $a(\tau,\theta_{\mathrm{low},e},f)$ of the amplitude $a_{\mathrm{low}}(\tau)$ of the mode v_i and a correction $\delta\theta(\tau,\theta_{\mathrm{low},e},f)$ determining an updated estimate $\theta_{\mathrm{low},e}(\tau)+\delta\theta(\tau,\theta_{\mathrm{low},e},f)$ of the estimated mode phase function $\theta_{\mathrm{low},e}$.

Supposing $\alpha>0$, $\tau\in[-1,1]$, and $n\in\{0,\dots,d\}$, let $\chi_{n,c}^{\tau,\theta_e}$ and $\chi_{n,s}^{\tau,\theta_e}$ be the wavelets defined by

$$\begin{aligned}\chi_{n,c}^{\tau,\theta_e}(t) &:= \cos(\theta_e(t))(t-\tau)^n e^{-\left(\frac{\dot{\theta}_e(\tau)(t-\tau)}{\alpha}\right)^2}\\ \chi_{n,s}^{\tau,\theta_e}(t) &:= \sin(\theta_e(t))(t-\tau)^n e^{-\left(\frac{\dot{\theta}_e(\tau)(t-\tau)}{\alpha}\right)^2},\end{aligned}\tag{4.2.1}$$

and let ξ_{τ,θ_e} be the Gaussian process defined by

$$\xi_{\tau,\theta_e}(t):=\sum_{n=0}^{d}\left(X_{n,c}\chi_{n,c}^{\tau,\theta_e}(t)+X_{n,s}\chi_{n,s}^{\tau,\theta_e}(t)\right),\tag{4.2.2}$$

where $X_{n,c}, X_{n,s} \sim \mathcal{N}(0,1)$ are IID random variables. Let f_τ be the Gaussian windowed signal defined by

$$f_\tau(t) = e^{-\left(\frac{\dot{\theta}_e(\tau)(t-\tau)}{\alpha}\right)^2} f(t), \quad t \in [-1,1]\,, \tag{4.2.3}$$

and, for $(n,j) \in \{0,\dots,d\} \times \{c,s\}$, let

$$Z_{n,j}(\tau,\theta_e,f) := \lim_{\sigma \downarrow 0} \mathbb{E}\big[X_{n,j} \big| \xi_{\tau,\theta_e} + \xi_\sigma = f_\tau\big]\,, \tag{4.2.4}$$

where ξ_σ is white noise, independent of ξ_{τ,θ_e}, with variance σ^2. To compute $Z_{n,j}$, observe that since both ξ_{τ,θ_e} and ξ_σ are Gaussian fields, it follows from (1.4.1) that

$$\mathbb{E}\big[\xi_{\tau,\theta_e} \big| \xi_{\tau,\theta_e} + \xi_\sigma\big] = A_\sigma(\xi_{\tau,\theta_e} + \xi_\sigma)$$

for the linear mapping

$$A_\sigma = Q_{\tau,\theta_e}\big(Q_{\tau,\theta_e} + \sigma^2 I\big)^{-1}\,,$$

where $Q_{\tau,\theta_e} : L^2 \to L^2$ is the covariance operator of the Gaussian field ξ_{τ,θ_e} and $\sigma^2 I$ is the covariance operator of ξ_σ. Using the characterization of the limit of Tikhonov regularization as the Moore-Penrose inverse, see, e. g., Barata and Hussein [8, Thm. 4.3], along with the orthogonal projections connected with the Moore-Penrose inverse, we conclude that $\lim_{\sigma \to 0} A_\sigma = P_{\chi^{\tau,\theta_e}}$, where $P_{\chi^{\tau,\theta_e}}$ is the L^2-orthogonal projection onto the span $\chi^{\tau,\theta_e} := \operatorname{span}\{\chi^{\tau,\theta_e}_{n,c}, \chi^{\tau,\theta_e}_{n,s} : n = 0,\dots,d\}$, and therefore

$$\lim_{\sigma \to 0} \mathbb{E}\big[\xi_{\tau,\theta_e} \big| \xi_{\tau,\theta_e} + \xi_\sigma\big] = P_{\chi^{\tau,\theta_e}}(\xi_{\tau,\theta_e} + \xi_\sigma)\,. \tag{4.2.5}$$

Since the definition (4.2.2) can be written $\xi_{\tau,\theta_e} = \sum_{n,j} X_{n,j}\chi^{\tau,\theta_e}_{n,j}$, summing (4.2.4) and using (4.2.5), we obtain

$$\sum_{n,j} Z_{n,j}(\tau,\theta_e,f)\chi^{\tau,\theta_e}_{n,j}(t) = P_{\chi^{\tau,\theta_e}} f_\tau(t), \quad t \in [-1,1]\,. \tag{4.2.6}$$

Consider the vector function $Z(\tau,\theta_e,f) \in \mathbb{R}^{2d+2}$ with components $Z_{n,j}(\tau,\theta_e,f)$, the $2d+2$ dimensional Gaussian random vector X with components $X_{n,j}$, $(n,j) \in \{0,\dots,d\} \times \{c,s\}$, and the $(2d+2)\times(2d+2)$ matrix A^{τ,θ_e} defined by

$$A^{\tau,\theta_e}_{(n,j),(n',j')} := \langle \chi^{\tau,\theta_e}_{n,j}, \chi^{\tau,\theta_e}_{n',j'} \rangle_{L^2[-1,1]}\,. \tag{4.2.7}$$

Straightforward linear algebra along with (4.2.6) establish that the vector $Z(\tau,\theta_e,f)$ can be computed as the solution of the linear system

$$A^{\tau,\theta_e} Z(\tau,\theta_e,f) = b^{\tau,\theta_e}(f), \tag{4.2.8}$$

where $b^{\tau,\theta_e}(f)$ is the $\mathbb{R}^{2d+2}$ vector with components $b^{\tau,\theta_e}_{n,j}(f) := \langle \chi^{\tau,\theta_e}_{n,j}, f_\tau \rangle_{L^2}$. See sub-figures (1) and (2) of both the top and bottom of Figure 4.5 for illustrations of the windowed signal $f_\tau(t)$ and of its projection $\lim_{\sigma\downarrow 0}\mathbb{E}\big[\xi_{\tau,\theta_e}\big|\xi_{\tau,\theta_e}+\xi_\sigma = f_\tau\big]$ in (4.2.5) corresponding to the signals f displayed in Figures 4.2 and 4.3.

To apply these formulations to construct the module, suppose that the signal is a single mode

$$f(t) = a(t)\cos(\theta(t)),$$

so that

$$f_\tau(t) = e^{-\left(\frac{\dot\theta_e(\tau)(t-\tau)}{\alpha}\right)^2} a(t)\cos(\theta(t))\,, \tag{4.2.9}$$

and consider the modified function

$$\bar f_\tau(t) = e^{-\left(\frac{\dot\theta_e(\tau)(t-\tau)}{\alpha}\right)^2}\left(\sum_{n=0}^{d}\frac{a^{(n)}(\tau)}{n!}(t-\tau)^n\right)\cos(\theta(t)) \tag{4.2.10}$$

obtained by replacing the function a with the first $d+1$ terms of its Taylor series about τ. In what follows, we will use the expression $\approx$ to articulate an informal approximation analysis. It is clear that $\bar f_\tau \in \chi^{\tau,\theta_e}$ and, since $\frac{\alpha}{\dot\theta_0(\tau)}$ is small, that $\langle \chi^{\tau,\theta_e}_{n,j}, f_\tau - \bar f_\tau\rangle_{L^2} \approx 0, \forall (n,j)$ and therefore $P_{\chi^{\tau,\theta_e}} f_\tau \approx \bar f_\tau$, and therefore (4.2.6) implies that

$$\sum_{j'} Z_{0,j'}(\tau,\theta_e,f)\chi^{\tau,\theta_e}_{0,j'}(t) \approx \bar f_\tau(t), \quad t\in[-1,1]\,, \tag{4.2.11}$$

which by (4.2.10) implies that

$$\sum_{j'} Z_{0,j'}(\tau,\theta_e,f)\chi^{\tau,\theta_e}_{0,j'}(t) \approx e^{-\left(\frac{\dot\theta_e(\tau)(t-\tau)}{\alpha}\right)^2} a(\tau)\cos(\theta(t)), \quad t\approx\tau\,, \tag{4.2.12}$$

which implies that

$$Z_{0,c}(\tau,\theta_e,f)\cos(\theta_e(t)) + Z_{0,s}(\tau,\theta_e,f)\sin(\theta_e(t)) \approx a(\tau)\cos(\theta(t)), \quad t\approx\tau\,. \tag{4.2.13}$$

Setting $\theta_\delta := \theta - \theta_e$ as the approximation error, using the cosine summation formula, we obtain

$$\begin{aligned} Z_{0,c}(\tau,\theta_e,f)\cos(\theta_e(t)) + Z_{0,s}(\tau,\theta_e,f)\sin(\theta_e(t)) \approx \\ a(\tau)\big(\cos(\theta_\delta(t))\cos(\theta_e(t)) - \sin(\theta_\delta(t))\sin(\theta_e(t)\big). \end{aligned} \tag{4.2.14}$$

However, $t\approx\tau$ implies that $\theta_\delta(t)\approx\theta_\delta(\tau)$, so that we obtain

$$\begin{aligned} Z_{0,c}(\tau,\theta_e,f)\cos(\theta_e(t)) + Z_{0,s}(\tau,\theta_e,f)\sin(\theta_e(t)) \approx \\ a(\tau)\big(\cos(\theta_\delta(\tau))\cos(\theta_e(t)) - \sin(\theta_\delta(\tau))\sin(\theta_e(t)\big)\,, \end{aligned} \tag{4.2.15}$$

which, since $\dot{\theta}_e(t)$ positive and bounded away from 0, implies that

$$\begin{aligned} Z_{0,c}(\tau,\theta_e,f) &\approx a(\tau)\cos(\theta_\delta(\tau)) \\ Z_{0,s}(\tau,\theta_e,f) &\approx -a(\tau)\sin(\theta_\delta(\tau))\,. \end{aligned}$$

Consequently, writing

$$\begin{aligned} a(\tau,\theta_e,f) &:= \sqrt{Z_{0,c}^2(\tau,\theta_e,f)+Z_{0,s}^2(\tau,\theta_e,f)} \\ \delta\theta(\tau,\theta_e,f) &:= \operatorname{atan2}\big(-Z_{0,s}(\tau,\theta_e,f),Z_{0,c}(\tau,\theta_e,f)\big)\,, \end{aligned} \tag{4.2.16}$$

we obtain that $a(\tau,\theta_e,f)\approx a(\tau)$ and $\delta\theta(\tau,\theta_e,f)\approx\theta_\delta(\tau)$. We will therefore use $a(\tau,\theta_e,f)$ to estimate the amplitude $a(\tau)$ of the mode corresponding to the estimate θ_e and $\delta\theta(\tau,\theta,f)$ to estimate the true mode phase θ through $\theta(\tau)=\theta_e(\tau)+\theta_\delta(\tau)\approx\theta_e(\tau)+\delta\theta(\tau,\theta_e,f)$. Unless otherwise specified, Equation (4.2.16) will take $d=2$. Experimental evidence indicates that $d=2$ is a sweet spot in the sense that $d=0$ or $d=1$ yields less fitting power, while larger d entails less stability. Iterating this refinement process will allow us to achieve near machine-precision accuracies in our phase/amplitude estimates. See sub-figures (1) and (2) of the top and bottom of Figure 4.6 for illustrations of $a(t)$, $a(\tau,\theta_e,v)(t)$, $\theta(t)-\theta_e(t)$ and $\delta\theta(\tau,\theta_e,v)(t)$ corresponding to the first mode v_1 of the signals v displayed in Figures 4.2.4 and 4.3.4.

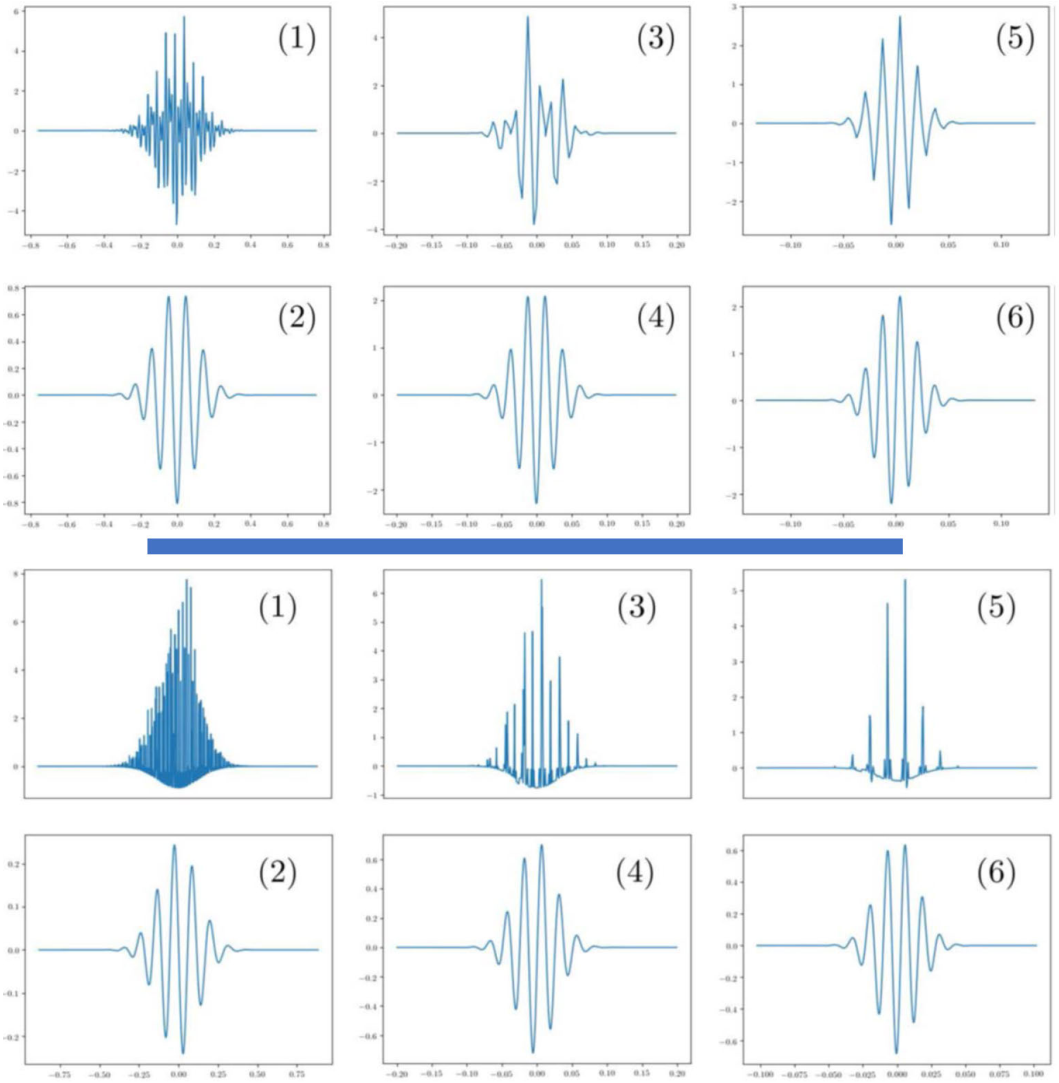

Figure 4.5: [102, Fig. 28], top: v is as in Figure 4.2 (the base waveform is triangular). Bottom: v is as in Figure 4.3 (the base waveform is EKG). Both top and bottom: $d = 2$, (1) The windowed signal v_τ (2) $\lim_{\sigma\downarrow 0}\mathbb{E}\left[\xi_{\tau,\theta_{1,e}}\middle|\xi_{\tau,\theta_{1,e}} + \xi_\sigma = v_\tau\right]$ (3) $(v - v_{1,e})_\tau$ (4) $\lim_{\sigma\downarrow 0}\mathbb{E}\left[\xi_{\tau,\theta_{2,e}}\middle|\xi_{\tau,\theta_{2,e}} + \xi_\sigma = (v - v_{1,e})_\tau\right]$ (5) $(v - v_{1,e} - v_{2,e})_\tau$ (6) $\lim_{\sigma\downarrow 0}\mathbb{E}\left[\xi_{\tau,\theta_{3,e}}\middle|\xi_{\tau,\theta_{3,e}} + \xi_\sigma = (v - v_{1,e} - v_{2,e})_\tau\right]$.

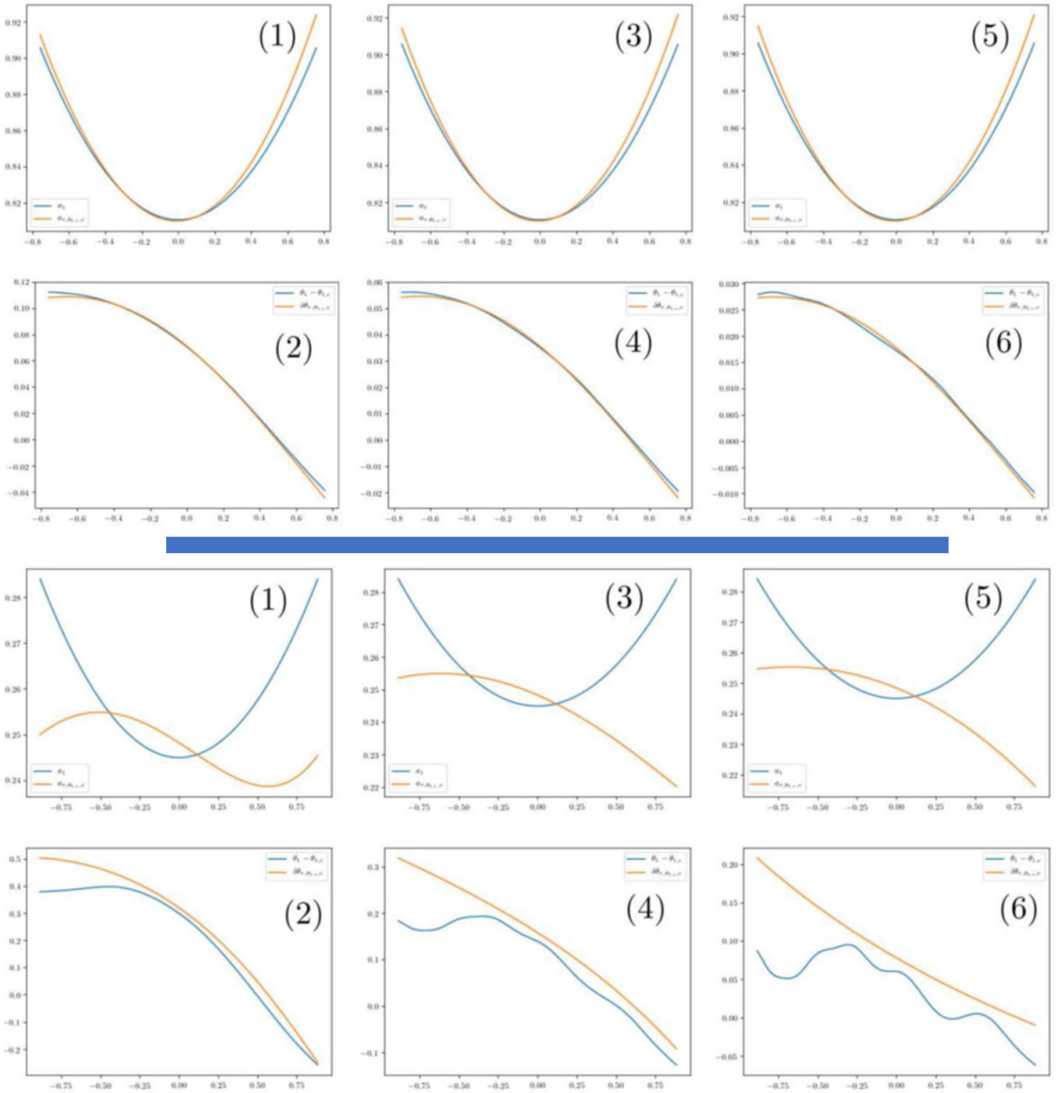

Figure 4.6: [102, Fig. 29], top: v is as in Figure 4.2 (the base waveform is triangular). Bottom: v is as in Figure 4.3 (the base waveform is EKG). Both top and bottom: $\tau = 0$. (1) the amplitude of the first mode $a_1(t)$ and its local Gaussian regression estimation $a(\tau, \theta_{1,e}, v)(t)$ (2) the error in estimated phase of the first mode $\theta_1(t) - \theta_{1,e}(t)$ and its local Gaussian regression $\delta\theta(\tau, \theta_{1,e}, v)(t)$ (3, 4) are as (1,2) with v and $\theta_{1,e}$ replaced by $v - v_{1,e}$ and $\theta_{2,e}$ (5,6) are as (1,2) with v and $\theta_{1,e}$ replaced by $v - v_{1,e} - v_{2,e}$ and $\theta_{3,e}$.

4.3 The iterated micro-local KMD algorithm.

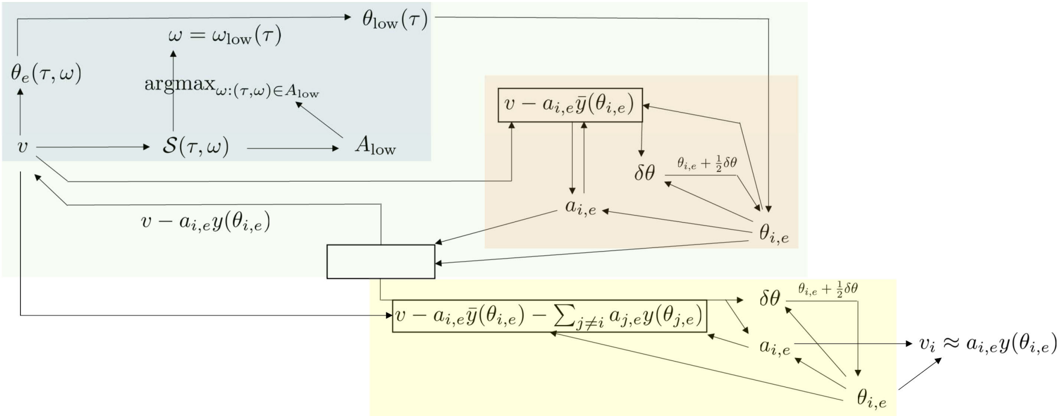
$\theta_{\text{low}}(\tau)$
$\omega = \omega_{\text{low}}(\tau)$
$\theta_e(\tau,\omega)$
$\text{argmax}_{\omega:(\tau,\omega)\in A_{\text{low}}}$
v
$\mathcal{S}(\tau,\omega)$
A_{low}
$v - a_{i,e}\bar{y}(\theta_{i,e})$
$\delta\theta$
$\theta_{i,e} + \frac{1}{2}\delta\theta$
$a_{i,e}$
$\theta_{i,e}$
$v - a_{i,e}y(\theta_{i,e})$
$v - a_{i,e}\bar{y}(\theta_{i,e}) - \sum_{j\neq i} a_{j,e}y(\theta_{j,e})$
$\delta\theta$
$\theta_{i,e} + \frac{1}{2}\delta\theta$
$a_{i,e}$
$v_i \approx a_{i,e}y(\theta_{i,e})$
$\theta_{i,e}$

Figure 4.7: [102, Fig. 27], modular representation of Algorithm 3, described in this section. The blue module represents the estimation of the lowest frequency of signal represented by v as illustrated in Figure 4.4. The brown module represents the iterative estimation of the mode with lowest instantaneous frequency of steps 10 through 14 of Algorithm 3. The yellow module represents the iterative refinement of all the modes in steps 21 through 28. The brown and yellow modules used to refine phase/amplitude estimates use the same code.

The method of estimating the lowest instantaneous frequency, described in Section 4.1, provides a foundation for the iterated micro-local KMD algorithm [102, Sec. 8.3], Algorithm 3. This algorithm is presented its modular representation in Figure 4.7, using Figures 4.4, 4.5, and 4.6. We begin by letting

$$y(t) = c_1\cos(t) + \sum_{n=2}^{\infty} c_n\cos(nt + d_n) \tag{4.3.1}$$

be the Fourier representation of the base waveform y (which, without loss of generality, has been shifted so that the first sine coefficient is zero) and write

$$\bar{y}(t) := y(t) - c_1\cos(t) \tag{4.3.2}$$

for its overtones.

Algorithm 3 Iterated micro-local KMD.

```
1:  $i \leftarrow 1$
2:  $v^{(1)} \leftarrow v$
3:  **while** true **do**
4:      **if** $\theta_{\text{low}}(v^{(i)}) = \emptyset$ **then**
5:          break loop
6:      **else**
7:          $\theta_{i,e} \leftarrow \theta_{\text{low}}(v^{(i)})$
8:      **end if**
9:      $a_{i,e}(t) \leftarrow 0$
10:     **repeat**
11:         **for** $j$ in $\{1,...,i\}$ **do**
12:             $v_{j,\text{res}} \leftarrow v - a_{j,e}\bar{y}(\theta_{j,e}) - \sum_{k\neq j,k\leq i} a_{k,e}y(\theta_{k,e})$
13:             $a_{j,e}(\tau)$[1] $\leftarrow a\big(\tau,\theta_{j,e},v_{j,\text{res}}\big)/c_1$
14:             $\theta_{j,e}(\tau) \leftarrow \theta_{j,e}(\tau) + \frac{1}{2}\delta\theta\big(\tau,\theta_{j,e},v_{j,\text{res}}\big)$
15:         **end for**
16:     **until** $\sup_{i,\tau}\big|\delta\theta\big(\tau,\theta_{i,e},v_{i,\text{res}}\big)\big| < \epsilon_1$
17:     $v^{(i+1)} \leftarrow v - \sum_{j\leq i} a_{j,e}y(\theta_{j,e})$
18:     $i \leftarrow i+1$
19: **end while**
20: $m \leftarrow i-1$
21: **if** refine_final = **True then**
22:     **repeat**
23:         **for** $i$ in $\{1,...,m\}$ **do**
24:             $v_{i,\text{res}} \leftarrow v - a_{i,e}\bar{y}(\theta_{i,e}) - \sum_{j\neq i} a_{j,e}y(\theta_{j,e})$
25:             $a_{i,e}(\tau) \leftarrow a\big(\tau,\theta_{i,e},v_{i,\text{res}}\big)$
26:             $\theta_{i,e}(\tau) \leftarrow \theta_{i,e}(\tau) + \frac{1}{2}\delta\theta\big(\tau,\theta_{i,e},v_{i,\text{res}}\big)$
27:         **end for**
28:     **until** $\sup_{j,\tau}\big|\delta\theta\big(\tau,\theta_{j,e},v_{j,\text{res}}\big)\big| < \epsilon_2$
29: **end if**
30: Return the modes $v_{i,e} \leftarrow a_{i,e}(t)y(\theta_{i,e}(t))$ for $i = 1,...,m$
```

Let us describe how steps 1 to 19 provide refined estimates for the amplitude and the phase of each mode $v_i, i \in \{1,\ldots,m\}$ of the signal v. Although the overtones of y prevent us from simultaneously approximating all the instantaneous frequencies $\dot{\theta}_i$ from the max-pool energy of the signal v, since the lowest mode $v_{\text{low}} = a_{\text{low}}y(\theta_{\text{low}})$ can be decomposed into the sum $v_{\text{low}} = a_{\text{low}}c_1\cos(\theta_{\text{low}}) + a_{\text{low}}\bar{y}(\theta_{\text{low}})$ of a signal $a_{\text{low}}c_1\cos(\theta_{\text{low}})$ with a cosine waveform plus the signal $a_{\text{low}}\bar{y}(\theta_{\text{low}})$ containing its higher frequency overtones, the method of Section 4.1 can be applied to obtain an

[1] All statements in Algorithms with dummy variable τ or t imply a loop over all values of τ in the mesh $\mathcal{T}$.

estimate $\theta_{\text{low},e}$ of θ_{low} and (4.2.16) can be applied to obtain an estimate $a_{\text{low},e}c_1$ of $a_{\text{low}}c_1$, producing an estimate $a_{\text{low},e}c_1\cos(\theta_{\text{low},e})$ of the primary component $a_{\text{low}}c_1\cos(\theta_{\text{low}})$ of the first mode. Since c_1 is known, this estimate produces the estimate $a_{\text{low},e}\bar{y}(\theta_{\text{low},e})$ for the overtones of the lowest mode. Recall that we calculate all quantities over the interval $[-1,1]$ in this setting. Estimates near the borders, -1 and 1, will be less precise but will be refined in the following loops. To improve the accuracy of this estimate, in steps 13 and 14 the micro-local KMD of Section 4.2 is iteratively applied to the residual signal of every previously identified mode $v_{j,\text{res}} \leftarrow v - a_{j,e}\bar{y}(\theta_{j,e}) - \sum_{k\neq j,k\leq i} a_{k,e}y(\theta_{k,e})$, consisting of the signal v with the estimated modes $k \neq j$ as well as the overtones of estimated mode j removed. This residual is the sum of the estimation of the isolated base frequency component of v_j and $\sum_{j>i} v_j$. The rate parameter $1/2$ in line 14 is to avoid overcorrecting the phase estimates, while the parameters ϵ_1 and ϵ_2 in steps 10 and 21 are pre-specified accuracy thresholds. The resulting estimated lower modes are then removed from the signal to determine the residual $v^{(i+1)} := v - \sum_{j\leq i} a_{j,e}y(\theta_{j,e})$ in line 17.

Iterating this process, we peel off an estimate $a_{i,e}y(\theta_{i,e})$ of the mode corresponding to the lowest instantaneous frequency of the residual $v^{(i)} := v - \sum_{j\leq i-1} a_{j,e}y(\theta_{j,e})$ of the signal v obtained in line 17, removing the interference of the first $i-1$ modes, including their overtones, in our estimate of the instantaneous frequency and phase of the i-th mode. See Figure 4.4 for the evolution of the A_{low} sausage as these modes are peeled off. See sub-figures (3) and (5) of the top and bottom of Figure 4.5 for the results of peeling off the first two estimated modes of the signal v corresponding to both Figures 4.2 and 4.3 and sub-figures (4) and (6) for the results of the corresponding projections in (4.2.5). See sub-figures (3) and (4) of the top and bottom of Figure 4.6 for amplitude and its estimate of the results of peeling off the first estimated mode and sub-figures (5) and (6) corresponding to peeling off the first two estimated modes of the signal v corresponding to both Figures 4.2 and 4.3.

After the amplitude/phase estimates $a_{i,e}, \theta_{i,e}, i \in \{1,\ldots,m\}$, have been obtained in steps 1 to 19, we have the option to further improve our estimates in a final optimization loop in steps 21 to 28. This choice is symbolized by variable "refine_final" which is **True** if we wish to run this final refinement, which enables us to achieve even higher accuracies by iterating the micro local KMD of Section 4.2 on the residual signals $v_{i,\text{res}} \leftarrow v - a_{i,e}\bar{y}(\theta_{i,e}) - \sum_{j\neq i} a_{j,e}y(\theta_{j,e})$, consisting of the signal v with all the estimated modes $j \neq i$ and estimated overtones of the mode i removed.

The proposed algorithm can be further improved by (1) applying a Savitsky-Golay filter to locally smooth (denoise) the curves corresponding to each estimate $\theta_{i,e}$ (which corresponds to refining our phase estimates through GPR filtering) (2) starting with a larger α (to decrease interference from other modes/overtones) and slowly reducing its value in the optional final refinement loop (to further localize our estimates after other components, and hence interference, have been mostly eliminated).

4.4 Numerical experiments

Here, we present results for both the triangle and EKG base waveform examples [102, Sec. 8.4]. As discussed in the previous section, these results are visually displayed in Figures 4.5 and 4.6.

Triangle wave example

The base waveform is the triangle wave displayed in Figure 4.1. We observe the signal v on a mesh spanning $[-1, 1]$ spaced at intervals of $\frac{1}{5000}$ and aim to recover each mode v_i over this time mesh. We take $\alpha = 25$ within the first refinement loop corresponding to steps 1 to 19 and slowly decreased it to 6 in the final loop corresponding to steps 22 to 28. The amplitudes and frequencies of each of the modes are shown in Figure 4.2. The recovery errors of each mode as well as their amplitude and phase functions over the whole interval $[-1, 1]$ and the interior third $[-\frac{1}{3}, \frac{1}{3}]$ are displayed in Tables 4.1 and 4.2, respectively. In the interior third of the interval, errors were found to be on the order of 10^{-9} for the first signal component and approximately 10^{-7} for the higher two. However, over the full interval, the corresponding figures are in the 10^{-4} and 10^{-3} ranges due to recovery errors near the boundaries, -1 and 1, of the interval. Still, a plot superimposing v_i and $v_{i,e}$ would visually appear to be one curve over $[-1, 1]$ due to the negligible recovery errors.

Mode	$\frac{\|v_{i,e}-v_i\|_{L^2}}{\|v_i\|_{L^2}}$	$\frac{\|v_{i,e}-v_i\|_{L^\infty}}{\|v_i\|_{L^\infty}}$	$\frac{\|a_{i,e}-a_i\|_{L^2}}{\|a_i\|_{L^2}}$	$\|\theta_{i,e}-\theta_i\|_{L^2}$
$i = 1$	5.47×10^{-4}	3.85×10^{-3}	2.80×10^{-4}	4.14×10^{-5}
$i = 2$	6.42×10^{-4}	2.58×10^{-3}	3.80×10^{-5}	1.85×10^{-4}
$i = 3$	5.83×10^{-4}	6.29×10^{-3}	2.19×10^{-4}	6.30×10^{-5}

Table 4.1: Signal component recovery errors in the triangle base waveform example over $[-1, 1]$.

Mode	$\frac{\|v_{i,e}-v_i\|_{L^2}}{\|v_i\|_{L^2}}$	$\frac{\|v_{i,e}-v_i\|_{L^\infty}}{\|v_i\|_{L^\infty}}$	$\frac{\|a_{i,e}-a_i\|_{L^2}}{\|a_i\|_{L^2}}$	$\|\theta_{i,e}-\theta_i\|_{L^2}$
$i=1$	1.00×10^{-8}	2.40×10^{-8}	7.08×10^{-9}	6.52×10^{-9}
$i=2$	2.74×10^{-7}	2.55×10^{-7}	1.87×10^{-8}	2.43×10^{-7}
$i=3$	2.37×10^{-7}	3.67×10^{-7}	1.48×10^{-7}	1.48×10^{-7}

Table 4.2: Signal component recovery errors in the triangle base waveform example over $[-\frac{1}{3},\frac{1}{3}]$.

EKG wave example

The base waveform is the EKG wave displayed in Figure 4.1. We use the same discrete mesh as in the triangle case. Here, we took $\alpha = 25$ in the loop corresponding to steps 1 to 19 and slowly decreased it to 15 in the final loop corresponding to steps 22 to 28. The amplitudes and frequencies of each of the modes are shown in Figure 4.3, while the recovery error of each mode as well as their amplitude and phase functions are shown both over the whole interval $[-1,1]$ and the interior third $[-\frac{1}{3},\frac{1}{3}]$ in Tables 4.3 and 4.4, respectively. Within the interior third of the interval, amplitude and phase relative errors are found to be on the order of 10^{-4} to 10^{-5} in this setting. However, over $[-1,1]$, the mean errors are more substantial, with amplitude and phase estimates in the 10^{-1} to 10^{-3} range. Note the high error rates in L^∞ stemming from errors in placement of the tallest peak (the region around which is known as the R wave in the EKG community). In the center third of the interval, $v_{i,e}$ and v_i are visually indistinguishable due to the small recovery errors.

Mode	$\frac{\|v_{i,e}-v_i\|_{L^2}}{\|v_i\|_{L^2}}$	$\frac{\|v_{i,e}-v_i\|_{L^\infty}}{\|v_i\|_{L^\infty}}$	$\frac{\|a_{i,e}-a_i\|_{L^2}}{\|a_i\|_{L^2}}$	$\|\theta_{i,e}-\theta_i\|_{L^2}$
$i=1$	5.66×10^{-2}	1.45×10^{-1}	4.96×10^{-3}	8.43×10^{-3}
$i=2$	4.61×10^{-2}	2.39×10^{-1}	2.35×10^{-2}	1.15×10^{-2}
$i=3$	1.34×10^{-1}	9.39×10^{-1}	9.31×10^{-3}	2.69×10^{-2}

Table 4.3: Signal component recovery errors on $[-1,1]$ in the EKG base waveform example.

Mode	$\frac{\|v_{i,e}-v_i\|_{L^2}}{\|v_i\|_{L^2}}$	$\frac{\|v_{i,e}-v_i\|_{L^\infty}}{\|v_i\|_{L^\infty}}$	$\frac{\|a_{i,e}-a_i\|_{L^2}}{\|a_i\|_{L^2}}$	$\|\theta_{i,e}-\theta_i\|_{L^2}$
$i=1$	1.80×10^{-4}	3.32×10^{-4}	3.52×10^{-5}	2.85×10^{-5}
$i=2$	4.35×10^{-4}	5.09×10^{-4}	3.35×10^{-5}	7.18×10^{-5}
$i=3$	3.63×10^{-4}	1.08×10^{-3}	7.23×10^{-5}	6.26×10^{-5}

Table 4.4: Signal component recovery errors on $[-\frac{1}{3},\frac{1}{3}]$ in the EKG base waveform example.

Chapter 5

ITERATED MICRO-LOCAL KERNEL MODE DECOMPOSITION FOR UNKNOWN BASE WAVEFORMS

We continue our discussion of KMD techniques by examining its application to an extension of original mode recovery problem, Problem 7. We generalize the problem to the case where base waveforms of each mode are unknown [102, Sec. 9] and is formally stated below in Problem 9. Previously, in Section 4, we discussed how GPR can be applied to learn the instantaneous amplitudes and phases of each mode. In the context of the unknown waveform problem, we will introduce *micro-local waveform KMD* in Section 5.1, which again utilizes GPR and is able to estimate waveforms of modes.

Problem 9. *For $m \in \mathbb{N}^*$, let $a_1, \dots, a_m$ be piecewise smooth functions on $[-1, 1]$, let $\theta_1, \dots, \theta_m$ be piecewise smooth functions on $[-1, 1]$ such that the instantaneous frequencies $\dot{\theta}_i$ are strictly positive and well separated, and let $y_1, \dots, y_m$ be square-integrable 2π-periodic functions. Assume that m and the a_i, θ_i, y_i are all unknown. Given the observation*

$$v(t) = \sum_{i=1}^{m} a_i(t) y_i\big(\theta_i(t)\big), \quad t \in [-1, 1], \tag{5.0.1}$$

recover the modes $v_i(t) := a_i(t) y_i\big(\theta_i(t)\big)$.

To avoid ambiguities caused by overtones with the unknown waveforms y_i, we will assume that the corresponding functions $(k\dot{\theta}_i)_{t\in[-1,1]}$ and $(k'\dot{\theta}_{i'})_{t\in[-1,1]}$ are distinct for $i \neq i'$ and $k, k' \in \mathbb{N}^*$, that is, they may be equal for some t but not for all t. We represent the i-th base waveform y_i through its Fourier series

$$y_i(t) = \cos(t) + \sum_{k=2}^{k_{\max}} \big(c_{i,(k,c)} \cos(kt) + c_{i,(k,s)} \sin(kt)\big), \tag{5.0.2}$$

that, without loss of generality, has been scaled and translated. Moreover, since we operate in a discrete setting, we also truncate the series at a finite level $k_{\max}$, which is naturally bounded by the inverse of the resolution of the discretization in time. To

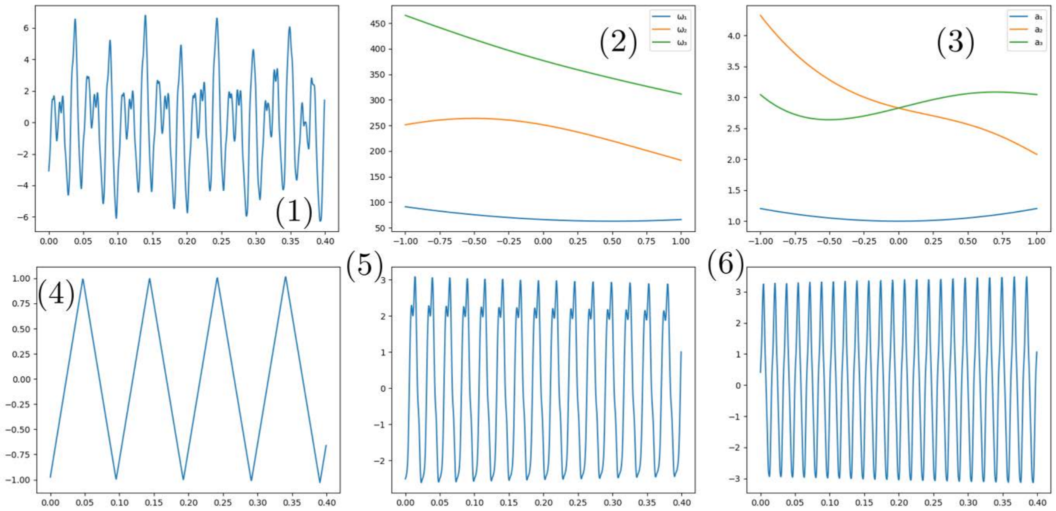

Figure 5.1: [102, Fig. 30], (1) signal v (the signal is defined over $[-1,1]$ but displayed over $[0,0.4]$ for visibility) (2) instantaneous frequencies $\omega_i := \dot{\theta}_i$ (3) amplitudes a_i (4, 5, 6) Modes v_1, v_2, v_3 over $[0,0.4]$ (mode plots have also been zoomed in for visibility).

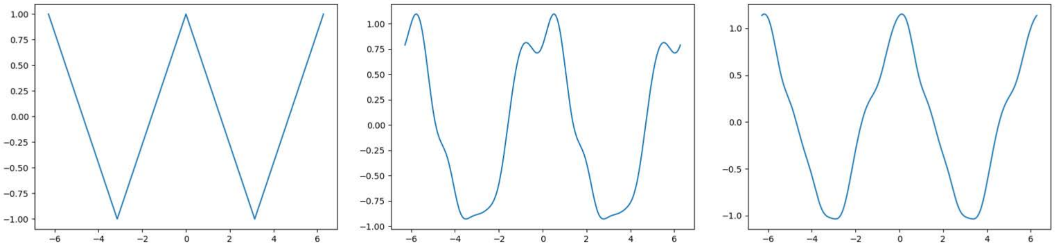

Figure 5.2: [102, Fig. 31], illustrations showing (1) y_1 (2) y_2 (3) y_3.

illustrate our approach, we consider the signal $v = v_1 + v_1 + v_3$ and its corresponding modes $v_i := a_i(t)y_i\big(\theta_i(t)\big)$ displayed in Figure 5.1, where the corresponding base waveforms y_1, y_2 and y_3 are shown in Figure 5.2 and described in Section 5.3.

5.1 Micro-local waveform KMD

We are now describing the micro-local *waveform* KMD [102, Sec. 9.1], Algorithm 4, which takes as inputs a time τ, estimated instantaneous amplitude and phase functions $t \rightarrow a(t), \theta(t)$, and a signal v, and outputs an estimate of the waveform $y(t)$ associated with the phase function θ. The proposed approach is a direct extension of the one presented in Section 4.2 and the shaded part of Figure 5.3 shows the new block which will be added to Algorithm 3, the algorithm designed for the case when waveforms are non-trigonometric and known. As described below this new block produces an estimator $y_{i,e}$ of the waveform y_i from an estimate $\theta_{i,e}$ of the phase θ_i.

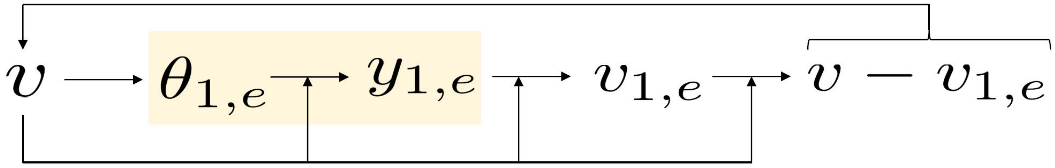

Figure 5.3: [102, Fig. 32], high level structure of Algorithm 4 for the case when the waveforms are unknown.

Given $\alpha > 0$, $\tau \in [-1, 1]$, and differentiable function $t \to \theta(t)$, define the Gaussian process

$$\xi^y_{\tau,\theta}(t) = e^{-\left(\frac{\dot{\theta}_e(\tau)(t-\tau)}{\alpha}\right)^2}\left(X^y_{1,c}\cos\left(\theta(t)\right) + \sum_{k=2}^{k_{\max}}\left(X^y_{k,c}\cos\left(k\theta(t)\right) + X^y_{k,s}\sin\left(k\theta(t)\right)\right)\right), \tag{5.1.1}$$

where $X^y_{1,c}$, $X^y_{k,c}$, and $X^y_{k,s}$ are independent $\mathcal{N}(0,1)$ random variables. Let

$$v_\tau(t) := e^{-\left(\frac{\dot{\theta}_e(\tau)(t-\tau)}{\alpha}\right)^2} v(t), \quad \tau \in [-1, 1], \tag{5.1.2}$$

be the windowed signal, and define

$$Z^y_{k,j}(\tau, \theta, v) := \lim_{\sigma \downarrow 0} \mathbb{E}\left[X^y_{k,j} \middle| \xi^y_{\tau,\theta} + \xi_\sigma = v_\tau\right], \tag{5.1.3}$$

and, for $k \in \{2, \ldots, k_{\max}\}$, $j \in \{c, s\}$, let

$$c_{k,j}(\tau, \theta, v) := \frac{Z^y_{k,j}(\tau, \theta, v)}{Z^y_{1,c}(\tau, \theta, v)}. \tag{5.1.4}$$

When the assumed phase function $\theta := \theta_{i,e}$ is close to the phase function θ_i of the i-th mode of the signal v in the expansion (5.0.1), $c_{k,j}(\tau, \theta_{i,e}, v)$ yields an estimate of the Fourier coefficient $c_{i,(k,j)}$ (5.0.2) of the i-th base waveform y_i at time $t = \tau$. This waveform recovery is susceptible to error when there is interference in the overtone frequencies (that is for the values of τ at which $j_1\dot{\theta}_{i_1} \approx j_2\dot{\theta}_{i_2}$ for $i_1 < i_2$). However, since the coefficient $c_{i,(k,j)}$ is independent of time, we can overcome this by computing $c_{k,j}(\tau, \theta_{i,e}, v)$ at each time τ and take the most common approximate value as follows. Let $T \subset [-1,1]$ be the finite set of values of τ in the numerical discretization of the time axis with $N := |T|$ elements. For interval $I \subset \mathbb{R}$,

$$T_I := \{\tau \in T | c_{k,j}(\tau, \theta_{i,e}, v) \in I\}\,, \tag{5.1.5}$$

and let $N_I := |T_I|$ denote the number of elements of T_I. Let $I_{\max}$ be a maximizer of the function $I \longrightarrow N_I$ over intervals of fixed width L, and define the estimate

$$c_{k,j}(\theta_{i,e}, v) := \begin{cases} \frac{1}{N_{I_{\max}}} \sum_{\tau \in T_{I_{\max}}} c_{k,j}(\tau, \theta_{i,e}, v) & , \quad \frac{N_{I_{\max}}}{N} \geq 0.05 \\ 0 & , \quad \frac{N_{I_{\max}}}{N} < 0.05 \end{cases}\,, \tag{5.1.6}$$

of the Fourier coefficient $c_{i,(k,j)}$ to be the average of the values of $c_{k,j}(\tau, \theta_{i,e}, v)$ over $\tau \in T_{I_{\max}}$. The interpretation of the selection of the cutoff 0.05 is as follows: if $\frac{N_{I_{\max}}}{N}$ is small then there is interference in the overtones at all time $[-1,1]$ and no information may be obtained about the corresponding Fourier coefficient. When the assumed phase function is near that of the lowest frequency mode v_1, which we write $\theta := \theta_{1,e}$, Figures 5.4.2 and 4 shows zoomed-in histograms of the functions $\tau \longrightarrow c_{(3,c)}(\tau, \theta_{1,e}, v)$ and $\tau \longrightarrow c_{(3,s)}(\tau, \theta_{1,e}, v)$ displayed in Figures 5.4.1 and 3.

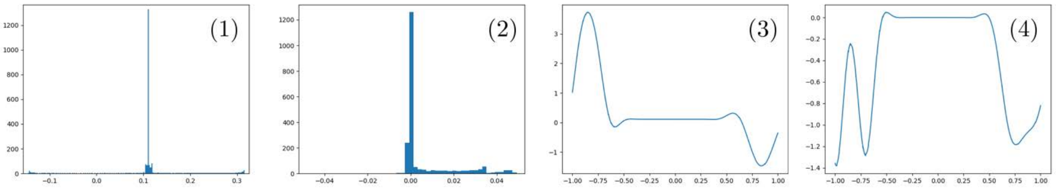

Figure 5.4: [102, Fig. 33], (1) a plot of the function $\tau \longrightarrow c_{(3,c)}(\tau, \theta_{1,e}, v)$ (2) a histogram (cropping outliers) with bin width 0.002 of $c_{(3,c)}(\tau, \theta_{1,e}, v)$ values. The true value $c_{1,(3,c)}$ is $1/9$ since y_1 is a triangle wave. (3) a plot of the function $\tau \longrightarrow c_{(3,s)}(\tau, \theta_{1,e}, v)$ (2) a histogram (cropping outliers) with bin width 0.002 of $c_{(3,s)}(\tau, \theta_{1,e}, v)$ values. The true value $c_{1,(3,s)}$ of this overtone is 0.

On the interval width L. In our numerical experiments, the recovered modes and waveforms show little sensitivity to the choice of L. In particular, we set L to be 0.002, whereas widths between 0.001 and 0.01 yield similar results. The rationale for the rough selection of the value of L is as follows. Suppose $v = \cos(\omega t)$ and $v' = v + \cos(1.5\omega t)$. Define the quantity

$$\max_{\tau}\big(c_{2,c}(\tau, \theta, v') - c_{2,c}(\tau, \theta, v)\big)\,, \tag{5.1.7}$$

with the intuition of approximating the maximum corruption by the $\cos(1.5\omega t)$ term in the estimated first overtone. This quantity provides a good choice for L and is mainly dependent on the selection of α and marginally on ω. For our selection of $\alpha = 10$, we numerically found its value to be approximately 0.002.

5.2 Iterated micro-local KMD with unknown waveforms algorithm

Algorithm 4 Iterated micro-local KMD with unknown waveforms.

```
$i \leftarrow 1$ and $v^{(1)} \leftarrow v$
while true do
    if $\theta_{\text{low}}(v^{(i)}) = \emptyset$ then
        break loop
    else
        $\theta_{i,e} \leftarrow \theta_{\text{low}}(v^{(i)})$
        $y_{i,e} \leftarrow \cos(t)$
    end if
    $a_{i,e}(t) \leftarrow 0$
    repeat
        for $l$ in $\{1, ..., i\}$ do
            $v_{l,\text{res}} \leftarrow v - a_{l,e}\bar{y}_{l,e}(\theta_{l,e}) - \sum_{k \neq l, k \leq i} a_{k,e} y_{l,e}(\theta_{k,e})$
            $a_{l,e}(\tau) \leftarrow a\big(\tau, \theta_{l,e}, v_{l,\text{res}}\big)/c_1$
            $\theta_{l,e}(\tau) \leftarrow \theta_{l,e}(\tau) + \frac{1}{2}\delta\theta\big(\tau, \theta_{l,e}, v_{l,\text{res}}\big)$
            $c_{l,(k,j),e} \leftarrow c_{k,j}\big(\theta_{l,e}, v_{l,\text{res}}\big)$
            $y_{l,e}(t) \leftarrow \cos(t) + \sum_{k=2}^{k_{\max}} \big(c_{l,(k,c),e}\cos(kt) + c_{l,(k,s),e}\sin(kt)\big)$
        end for
    until $\sup_{l,\tau}\big|\delta\theta\big(\tau, \theta_{l,e}, v_{l,\text{res}}\big)\big| < \epsilon_1$
    $v^{(i+1)} \leftarrow v - \sum_{j \leq i} a_{j,e} y_{i,e}(\theta_{j,e})$
    $i \leftarrow i + 1$
end while
$m \leftarrow i - 1$
if refine_final = True then
    repeat
        for $i$ in $\{1, \ldots, m\}$ do
            $v_{i,\text{res}} \leftarrow v - a_{i,e}\bar{y}_{i,e}(\theta_{i,e}) - \sum_{j \neq i} a_{j,e} y_{j,e}(\theta_{j,e})$
            $a_{i,e}(\tau) \leftarrow a\big(\tau, \theta_{i,e}, v_{i,\text{res}}\big)$
            $\theta_{i,e}(\tau) \leftarrow \theta_{i,e}(\tau) + \frac{1}{2}\delta\theta\big(\tau, \theta_{i,e}, v_{i,\text{res}}\big)$
            $c_{i,(k,j),e} \leftarrow c_{k,j}\big(\theta_{i,e}, v - \sum_{j \neq i} a_{j,e} y_{j,e}(\theta_{j,e})\big)$
            $y_{i,e}(t) \leftarrow \cos(t) + \sum_{k=2}^{k_{\max}} \big(c_{i,(k,c),e}\cos(kt) + c_{i,(k,s),e}\sin(kt)\big)$
        end for
    until $\sup_{i,\tau}\big|\delta\theta\big(\tau, \theta_{i,e}, v_{i,\text{res}}\big)\big| < \epsilon_2$
end if
Return the modes $v_{i,e} \leftarrow a_{i,e}(t) y(\theta_{i,e}(t))$ for $i = 1, ..., m$
```

Except for the steps discussed in Section 5.1, Algorithm 4 [102, Sec. 9.2] is identical to Algorithm 3. As illustrated in Figure 5.3, we first identify the lowest frequency of the cosine component of each mode (steps 6 and 7 in Algorithm 4). Next, from steps 10 to 18, we execute a similar refinement loop as in Algorithm 3 with the addition of an application of micro-local waveform KMD on steps 15 and 16 to estimate base waveforms. Finally, once each mode has been identified, we again apply waveform estimation in steps 29-30 (after nearly eliminating other modes and reducing interference in overtones for higher accuracies).

5.3 Numerical experiments

To illustrate this learning of the base waveform of each mode, we take $v(t) = \sum_{i=1}^{3} a_i(t)y_i(\theta_i(t))$, where the lowest frequency mode $a_1(t)y_1(\theta_1(t))$ has the (unknown) triangle waveform y_1 of Figure 4.1 [102, Sec. 9.3]. We determine the waveforms $y_i, i = 2, 3$, randomly by setting $c_{i,(k,j)}$ to be zero with probability $1/2$ or to be a random sample from $\mathcal{N}(0, 1/k^4)$ with probability $1/2$, for $k \in \{2, \dots, 7\}$ and $j \in \{c, s\}$. The waveforms y_1, y_2, y_3 thus obtained are illustrated in Figure 5.2. The modes v_1, v_2, v_3, their amplitudes and instantaneous frequencies are shown in Figure 5.1.

Mode	$\frac{\lVert v_{i,e}-v_i\rVert_{L^2}}{\lVert v_i\rVert_{L^2}}$	$\frac{\lVert v_{i,e}-v_i\rVert_{L^\infty}}{\lVert v_i\rVert_{L^\infty}}$	$\frac{\lVert a_{i,e}-a_i\rVert_{L^2}}{\lVert a_i\rVert_{L^2}}$	$\lVert\theta_{i,e}-\theta_i\rVert_{L^2}$	$\frac{\lVert y_{i,e}-y_i\rVert_{L^2}}{\lVert y_i\rVert_{L^2}}$
$i = 1$	6.31×10^{-3}	2.39×10^{-2}	9.69×10^{-5}	1.41×10^{-5}	6.32×10^{-3}
$i = 2$	3.83×10^{-4}	1.08×10^{-3}	5.75×10^{-5}	1.16×10^{-4}	3.76×10^{-4}
$i = 3$	3.94×10^{-4}	1.46×10^{-3}	9.53×10^{-5}	6.77×10^{-5}	3.80×10^{-4}

Table 5.1: Signal component recovery errors over $[-1, 1]$ when the base waveforms are unknown

We use the same mesh and the same value of α values as in Section 4.4. The main source of error for the recovery of the first mode's base waveform stems from the fact that a triangle wave has an infinite number of overtones, while in our implementation, we estimate only the first 15 overtones. Indeed, the L^2 recovery error of approximating the first 16 tones of the triangle wave is 3.57×10^{-4}, while the full recovery errors are presented in Table 5.1. We omitted the plots of the $y_{i,e}$ as they are visually indistinguishable from those of the y_i. Note that errors are only slightly improved away from the borders as the majority of it is accounted for by the waveform recovery error.

5.4 Further work in kernel mode decomposition

Micro-local kernel mode decomposition is also shown to produce mode decomposition in cases with modes with crossing frequencies or vanishing amplitudes and noisy observations. This setting is summarized by Problem 10. Further, an illustrative example is given in Example 5.4.1 and Figure 5.5

Problem 10. *For $m \in \mathbb{N}^*$, let $a_1, \dots, a_m$ be piecewise smooth functions on $[-1, 1]$, and let $\theta_1, \dots, \theta_m$ be strictly increasing functions on $[-1, 1]$ such that, for $\epsilon > 0$ and $\delta \in [0, 1)$, the length of t with $\dot{\theta}_i(t)/\dot{\theta}_j(t) \in [1-\epsilon, 1+\epsilon]$ is less than δ. Assume that m and the a_i, θ_i are unknown, and the square-integrable 2π-periodic base waveform y is known. Given the observation $v(t) = \sum_{i=1}^{m} a_i(t) y\big(\theta_i(t)\big) + v_\sigma(t)$ (for $t \in [-1, 1]$), where v_σ is a realization of white noise with variance σ^2, recover the modes $v_i(t) := a_i(t) y\big(\theta_i(t)\big)$.*

Example 5.4.1. *Consider the problem of recovering the modes of the signal $v = v_1 + v_2 + v_3 + v_\sigma$ shown in Figure 5.5. Each mode has a triangular base waveform. In this example v_3 has the highest frequency and its amplitude vanishes over $t > -0.25$. The frequencies of v_1 and v_2, cross around $t = 0.25$. $v_\sigma \sim \mathcal{N}(0, \sigma^2 \delta(t-s))$ is white noise with standard deviation $\sigma = 0.5$. While the signal-to-noise ratio is $\mathrm{Var}(v_1 + v_2 + v_3)/\mathrm{Var}(v_\sigma) = 13.1$, the SNR ratio against each of the modes $\mathrm{Var}(v_i)/\mathrm{Var}(v_\sigma)$, $i = 1, 2, 3$, is 2.7, 7.7, and 10.7, respectively.*

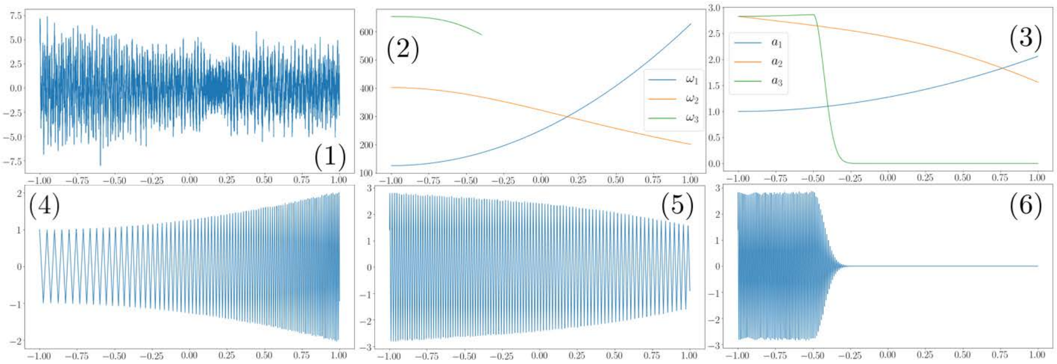

Figure 5.5: [102, Fig. 34], (1) signal v (2) instantaneous frequencies $\omega_i := \dot{\theta}_i$ (3) amplitudes a_i (4, 5, 6) modes v_1, v_2, v_3.

On a high level, the problem is approached by iterating the following process. During the life of the algorithm, the sets $\mathcal{V}$ and $\mathcal{V}_{\text{seg}}$ are maintained, containing the identified full modes and mode segments (which will be defined below), respectively.

First, we identify the lowest instantaneous frequency, $\omega_{\text{low}}(\tau)$ at each $\tau \in [-1, 1]$. Due to mode instantaneous frequency crossings and amplitude vanishings, this could correspond to multiple modes. We also determine the continuity of $\theta_{\text{low}}(\tau)$ and $\omega_{\text{low}}(\tau)$, and cut the domain at discontinuities. This leads to *mode fragments* (which are supported in between the domain cuts), which correspond to modes potentially only identified over a subset of domain $[-1, 1]$. It is checked whether each mode fragment can be extended with continuous θ_{low} or ω_{low} and is extended if possible, leading to *mode segments*. Then, the user then has the option whether to disregard, join, or pass on segments to the next iteration. The sets $\mathcal{V}$ and $\mathcal{V}_{\text{seg}}$ are updated accordingly. The identified modes in $\mathcal{V}$ are refined with a micro-local KMD loop. Finally, the identified modes and mode segments are peeled from the signal and we iterate the algorithm with the peeled signal. Further details, examples, and results are presented in [102, Sec. 10].

Chapter 6

KERNEL FLOWS

As introduced in [103], the Kernel Flow (KF) algorithm is a method for kernel selection/design in kriging/Gaussian Process Regression (GPR). It operates on the principle that a kernel is a *good* model of data if it is able to accurately make predictions on one subset of the data by observing another subset. We consider the supervised learning problem that approximates an unknown function u mapping Ω to $\mathbb{R}$ based on the input/output dataset $(t^i, y^i)_{1\leq i\leq N}$ (where $u(t^i) = y^i$). We define the vectors of input and output data as $\mathbf{D} = (t^i)_i \in \Omega^N$ and $\mathbf{Y} = (y^i)_i \in \mathbb{R}^N$. Any non-degenerate kernel $k(t, t')$ can be used to approximate u with the interpolant

$$u^\dagger(t) = \mathbf{k}(t, \mathbf{D})\mathbf{K}^{-1}\mathbf{Y}\,, \tag{6.0.1}$$

writing $\mathbf{k}(t, \mathbf{D}) = (k(t, t^i))_i \in \mathbb{R}^{1\times N}$, $\mathbf{K} = (k(t^i, t^j))_{i,j} \in \mathbb{R}^{N\times N}$. The kernel selection problem concerns the identification of a good kernel for performing this interpolation for a particular dataset. The KF algorithm's approach to this problem is to use the loss of accuracy incurred by removing half of the dataset as a loss of kernel selection. We will present a pair of variants of the KF algorithm, *parametric* and *non-parametric*, beginning with the former [103, Sec. 4], which is outlined in Algorithm 5.

6.1 Parametric KF Algorithm

Algorithm 5 Parametric KF Algorithm

```
Input: dataset $(t^i, y^i)_{1\leq i\leq N}$, kernel family $k_{\boldsymbol{\theta}}$, initial parameter $\boldsymbol{\theta}_0$
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_0$
repeat
    Randomly select $\{s_f(1), \dots s_f(N_f)\}$ from $\{1, \dots, N\}$ without replacement.

    $\mathbf{D^f} \leftarrow (t^{s_f(i)})_{1\leq i\leq N_c}$ and $\mathbf{Y^f} \leftarrow (y^{s_f(i)})_{1\leq i\leq N_c}$.
    Randomly select $\{s_c(1), \dots s_c(N_c)\}$ from $\{s_f(1), \dots s_f(N_f)\}$ without replacement.
    $\mathbf{D^c} \leftarrow (t^{s_c(i)})_{1\leq i\leq N_c}$ and $\mathbf{Y^c} \leftarrow (y^{s_c(i)})_{1\leq i\leq N_c}$.
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon \boldsymbol{\nabla}_{\boldsymbol{\theta}}\rho(\boldsymbol{\theta}, \mathbf{X^f}, \mathbf{Y^f}, \mathbf{X^c}, \mathbf{Y^c})$
until End criterion
Return optimized kernel parameter $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}$
```

As inputs, in step 1, the parametric variant of the KF algorithm takes a training dataset $(t^i, y^i)_i$, a parametric family of kernels $k_{\boldsymbol{\theta}}$, and a initial parameter $\boldsymbol{\theta}_0$. Kernel parameter $\boldsymbol{\theta}$ is first initialized to $\boldsymbol{\theta}_0$ in step 2. The main iterative process follows between steps 3 and 9, beginning with the random selection of size N_f subvectors $\mathbf{D^f}$ and $\mathbf{Y^f}$ of $\mathbf{D}$ and $\mathbf{Y}$ (through uniform sampling without replacement in the index set $\{1, \ldots, N\}$) as in steps 4 and 5. This selection is randomly sampled further with length N^c subvectors $\mathbf{D^c}$ and $\mathbf{Y^c}$ of $\mathbf{D^f}$ and $\mathbf{Y^f}$ (by selecting, at random, uniformly and without replacement, N_c of the indices defining D^f) in steps 6 and 7.

Finally in step 8, we update the kernel parameter $\boldsymbol{\theta}$ according to gradient descent with loss ρ. We define $\rho(\boldsymbol{\theta}, \mathbf{D^f}, \mathbf{Y^f}, \mathbf{D^c}, \mathbf{Y^c})$ to be the squared relative error (in the RKHS norm[1] $\|\cdot\|_{k_{\boldsymbol{\theta}}}$ defined by $k_{\boldsymbol{\theta}}$) between the interpolants $u^{\dagger,f}$ and $u^{\dagger,c}$ obtained from the two nested subsets of the dataset and the kernel $k_{\boldsymbol{\theta}}$, i.e.[2]

$$\rho(\boldsymbol{\theta}, \mathbf{X^f}, \mathbf{Y^f}, \mathbf{X^c}, \mathbf{Y^c}) := \frac{\|u^{\dagger,f} - u^{\dagger,c}\|^2_{k_{\boldsymbol{\theta}}}}{\|u^{\dagger,f}\|^2_{k_{\boldsymbol{\theta}}}} = 1 - \frac{\mathbf{Y}^{\mathbf{c},\top}\mathbf{k}_{\boldsymbol{\theta}}(\mathbf{X^c}, \mathbf{X^c})^{-1}\mathbf{Y_c}}{\mathbf{Y}^{\mathbf{f},\top}\mathbf{k}_{\boldsymbol{\theta}}(\mathbf{X^f}, \mathbf{X^f})^{-1}\mathbf{Y^f}}\,. \tag{6.1.1}$$

Note that loss ρ is doubly randomized through the selection of batches $(\mathbf{D^f}, \mathbf{Y^f})$ and sub-batches $(\mathbf{D^c}, \mathbf{Y^c})$. The gradient of ρ with respect to $\boldsymbol{\theta}$ is then computed and $\boldsymbol{\theta}$ is updated $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \delta\boldsymbol{\nabla}_{\boldsymbol{\theta}}\rho$. This process is iterated until an ending condition is satisfied, such as the number of iteration steps or $\rho < \rho_{\text{stop}}$. The optimized kernel parameter $\boldsymbol{\theta}^*$ is returned. The corresponding kernel, $k_{\boldsymbol{\theta}^*}$, can then be used to interpolate testing points. This algorithm is a stochastic gradient descent algorithm of ρ, hence the KF algorithm is a stochastic gradient descent algorithm.

The l_2-norm variant. In the application of the KF algorithm to NN, we will consider the l_2-norm variant of this algorithm (introduced in [103, Sec. 10]) in which the instantaneous loss ρ in (6.1.1) is replaced by the error (let $\|\cdot\|_2$ be the Euclidean l_2 norm) $e_2 := \|\mathbf{Y^f} - u^{\dagger,c}(\mathbf{D^f})\|_2^2$ of $u^{\dagger,c}$ in predicting the labels $\mathbf{Y^f}$, i.e.

$$e_2(\boldsymbol{\theta}, \mathbf{D^f}, \mathbf{Y^f}, \mathbf{D^c}, \mathbf{Y^c}) := \|\mathbf{Y^f} - \mathbf{k}_{\boldsymbol{\theta}}(\mathbf{D^f}, \mathbf{D^c})\mathbf{k}_{\boldsymbol{\theta}}(\mathbf{D^c}, \mathbf{D^c})^{-1}\mathbf{Y^c}\|_2^2\,. \tag{6.1.2}$$

A simple PDE model

To motivate, illustrate and study the parametric KF algorithm, it is useful to start with an application [103, Sec. 5] to the following simple PDE model amenable to

[1] Note that convergence in RKHS norm implies pointwise convergence.

[2] Where $u^{\dagger,f}(t) = \mathbf{k}_{\boldsymbol{\theta}}(t, \mathbf{X^f})\mathbf{k}_{\boldsymbol{\theta}}(\mathbf{X^f}, \mathbf{X^f})^{-1}\mathbf{Y^f}$ and $u^{\dagger,c}(t) = \mathbf{k}_{\boldsymbol{\theta}}(t, \mathbf{X^c})K_{\boldsymbol{\theta}}(\mathbf{X^c}, \mathbf{X^c})^{-1}\mathbf{Y^c}$. Further, ρ admits the representation on the left-hand side of equation (6.1.1) enabling its computation [103, Prop. 3.1].

detailed analysis [101]. Let u be the solution of second order elliptic PDE

$$\begin{cases} -\operatorname{div}\left(a(x)\nabla u(x)\right) = f(x) \quad x \in \Omega; \\ u = 0 \quad \text{on} \quad \partial\Omega\,, \end{cases} \tag{6.1.3}$$

with interval $\Omega \subset \mathbb{R}^1$ and uniformly elliptic symmetric matrix a with entries in $L^\infty(\Omega)$. We write $\mathcal{L} := -\operatorname{div}(a\nabla\cdot)$ for the corresponding linear bijection from $\mathcal{H}_0^1(\Omega)$ to $\mathcal{H}^{-1}(\Omega)$. In this proposed simple application we seek to both estimate conductivity coefficient a and recover the solution of (6.1.3) from the data $(x_i, y_i)_{1\le i\le N}$ and the information $u(x_i) = y_i$. In this example, we use kernel family, $\{G_b | b \in \mathcal{L}^\infty(\Omega), \operatorname{essinf}_\Omega(b) > 0\}$, where each G_b is the Green's function of operator $-\operatorname{div}(b\nabla\cdot)$. It is known that kernel recovery with G_a is minimax optimal in $\|\cdot\|$ norm [101]. In what follows, we will numerically demonstrate that the KF algorithm applied to this problem recovers a kernel G_{b^*} such that $a \approx b^*$.

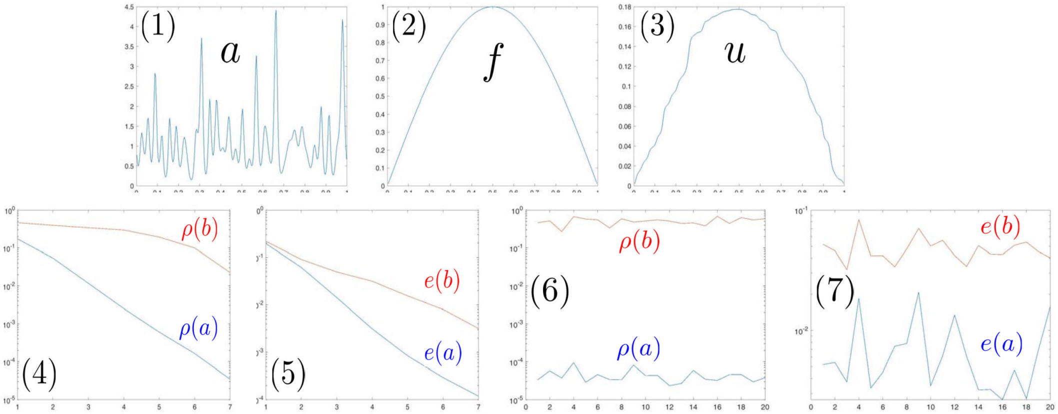

Figure 6.1: [103, Fig. 5], (1) a (2) f (3) u (4) $\rho(a)$ and $\rho(b)$ (where $b \equiv 1$) vs k (5) $e(a)$ and $e(b)$ vs k (6) 20 random realizations of $\rho(a)$ and $\rho(b)$ (7) 20 random realizations of $e(a)$ and $e(b)$.

Fig. 6.1 provides a numerical illustration of setting of our example, where Ω is discretized over 2^8 equally spaced interior points (and piecewise linear tent finite elements) and Fig. 6.1.1-3 shows a, f and u. For $k \in \{1,\dots,8\}$ and $i \in \mathcal{I}^{(k)} := \{1,\dots,2^k-1\}$ let $x_i^{(k)} = i/2^k$ and write $v_b^{(k)}$ for the interpolation of the data $(x_i^{(k)}, u(x_i^{(k)}))_{i\in\mathcal{I}^{(k)}}$ using the kernel G_b (note that $v_b^{(8)} = u$). Let $\|v\|_b$ be the energy norm $\|v\|_b^2 = \int_\Omega (\nabla v)^T b \nabla v$. Take $b \equiv 1$. Fig. 6.1.4 shows (in semilog scale) the values of $\rho(a) = \frac{\|v_a^{(k)} - v_a^{(8)}\|_a^2}{\|v_a^{(8)}\|_a^2}$ and $\rho(b) = \frac{\|v_b^{(k)} - v_b^{(8)}\|_b^2}{\|v_b^{(8)}\|_b^2}$ vs k. Note that the value of ratio ρ is much smaller when the kernel G_a is used for the interpolation of the data. The geometric decay $\rho(a) \le C2^{-2k}\frac{\|f\|_{L^2(\Omega)}}{\|u\|_a^2}$ is well known and has been extensively studied in Numerical Homogenization [101].

Fig. 6.1.5 shows (in semilog scale) the values of the prediction errors $e(a)$ and $e(b)$ (vs k) defined (after normalization) to be proportional to $\|v_a^{(k)}(x) - u(x)\|_{L^2(\Omega)}$ and $\|v_b^{(k)}(x) - u(x)\|_{L^2(\Omega)}$. Note again that the prediction error is much smaller when the kernel G_a is used for the interpolation.

Now, let us consider the case where the interpolation points form a random subset of the discretization points. Take $N_f = 2^7$ and $N_c = 2^6$. Let $\{x_1, \ldots, x_{N_f}\}$ be a subset with N_f distinct points of (the discretization points) $\{i/2^8 | i \in \mathcal{I}^{(8)}\}$ sampled with uniform distribution. Let $\{z_1, \ldots, z_{N_c}\}$ be a subset of N_c distinct points of X sampled with uniform distribution. Write v_b^f for the interpolation of the data $(x_i, u(x_i))$ using the kernel G_b and write v_b^c for the interpolation of the data $(z_i, u(z_i))$ using the kernel G_b. Fig. 6.1.6 shows in (semilog scale) 20 independent random realizations[3] of the values of $\rho(a) = \|v_a^f - v_a^c\|_a^2/\|v_a^f\|_a^2$ and $\rho(b) = \|v_b^f - v_b^c\|_b^2/\|v_b^f\|_b^2$. Fig. 6.1.7 shows in (semilog scale) 20 independent random realizations of the values of the prediction errors $e(a) \propto \|u - v_a^c\|_{L^2(\Omega)}$ and $e(b) \propto \|u - v_b^c\|_{L^2(\Omega)}$. Note again that the values of $\rho(a), e(a)$ are consistently and significantly lower than those of $\rho(b), e(b)$.

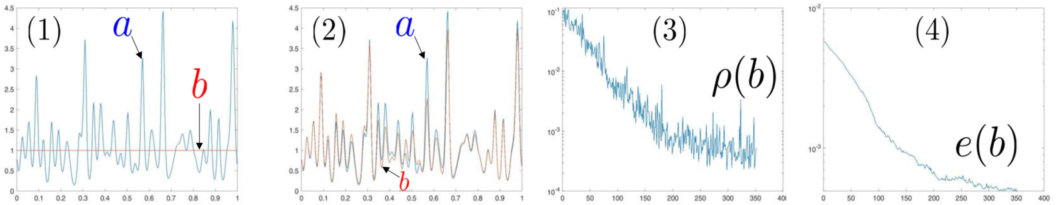

Figure 6.2: [103, Fig. 6], (1) a and b for $n = 1$ (2) a and b for $n = 350$ (2) $\rho(b)$ vs n (4) $e(b)$ vs n.

Fig. 6.2 provides a numerical illustration of an implementation of Alg. 5 with $N = N_f = 2^7$, $N_c = 2^6$, and n indexing each iteration of the KF algorithm starting with $n = 1$. In this implementation a, f and u are as in Fig. 6.1.1-3. The training data corresponds to N points $X = \{x_1, \ldots, x_N\}$ uniformly sampled (without replacement) from $\{i/2^8 | i \in \mathcal{I}^{(8)}\}$. Note that X remains fixed and since $N = N_f$, the larger batch

[3] Random realizations of the subsets.

(as in step 4 in Alg. 5) is always selected as X. The purpose of the algorithm is to learn the kernel G_a in the set of kernels $\{G_{b_W}|W\}$ parameterized by the vector W via

$$\log b_W = \sum_{i=1}^{2^6} (W_i^c \cos(2\pi i x) + W_i^s \sin(2\pi i x)) \,. \tag{6.1.4}$$

Using n to label its progression, Alg. 5 is initialized at $n = 1$ with the guess $b_0 \equiv 1$ (i.e., $W_0 \equiv 0$) (Fig. 6.2.1). Fig. 6.2.2 shows the value of b after $n = 350$ iterations and can be seen to approximate a. Fig. 6.2.3 shows the value of $\rho(b)$ vs n. Fig. 6.2.4 shows the value of the prediction error $e(b) \propto \|u - v_b^c\|_{L^2(\Omega)}$ vs n. The lack of smoothness of the plots of $\rho(b), e(b)$ vs n originate from the re-sampling of the set Z at each step n. Further details of this application of the KF algorithm can be found in [103, Sec. 5].

6.2 Non-parametric kernel flows

Recall that the parametric variant of the KF algorithm utilizes a parameterized family of kernels $k_{\boldsymbol{\theta}}$ and optimizes interpolation accuracy, ρ, with respect to $\boldsymbol{\theta}$. The interpolation returned by the algorithm is then $k_{\boldsymbol{\theta}^*}$ where $\boldsymbol{\theta}^*$ is the optimized kernel parameter. In contrast, the non-parametric version [103, Sec. 6] is initialized with kernel k and learns kernel of the form $k_F(t,t') = k(F(t), F(t'))$ where $F : \Omega \to \Omega$ is an arbitrary function. In this case,

$$\rho(F, \mathbf{D^f}, \mathbf{Y^f}, \mathbf{D^c}, \mathbf{Y^c}) := 1 - \frac{\mathbf{Y}^{\mathbf{c},\top}\mathbf{k}_F(\mathbf{X^c}, \mathbf{X^c})^{-1}\mathbf{Y_c}}{\mathbf{Y}^{\mathbf{f},\top}\mathbf{k}_F(\mathbf{X^f}, \mathbf{X^f})^{-1}\mathbf{Y^f}} \tag{6.2.1}$$

is optimized with respect to F. In practice, this is done by learning the ρ minimizing optimal deformations to training inputs in $\mathbf{D^f}$ and using kernel interpolation. This leads to a deformation G_n, which is used to update F at each iteration according to $(I + \epsilon G_n) \circ F$ where I is the identity function. Further mathematical detail can be found in [103, Sec. 6]. We will next present numerical examples of the non-parametric KF algorithm.

The Swiss Roll cheesecake example

We examine the application of the KF algorithm to the Swiss Roll cheesecake [103, Sec. 7], as illustrated in Figure 6.3.1. The dataset inputs lie in $\Omega = \mathbb{R}^2$ in the shape of two concentric spirals. Red points have label -1 and blue points have label 1. The purpose of this exposition is to illustrate the flow and deformations in each point in dataset. Hence, to do so, all datapoints will be considered as training points and we will not introduce a testing dataset. We select $N_f = N$ and $N_c = N_f/2$ and initialize the algorithm with

$$k(x, x') = e^{-\gamma\|x-x'\|^2} + \sigma^2\delta(x - x') . \tag{6.2.2}$$

Figure 6.3 shows the KF flow[4] of each of the $N = 250$ Swiss Roll points at different stages of the algorithm. We observe the concentric spirals being unrolled with all red and blue points placed in linearly separable regions. We also show differential fields of the deformations at different stages of the optimization of F_n in Figure 6.4. Further information on this example can be found in [103, Sec. 7] including potential instabilities of the algorithm when using a kernel without nugget $\sigma^2\delta(x - x')$.

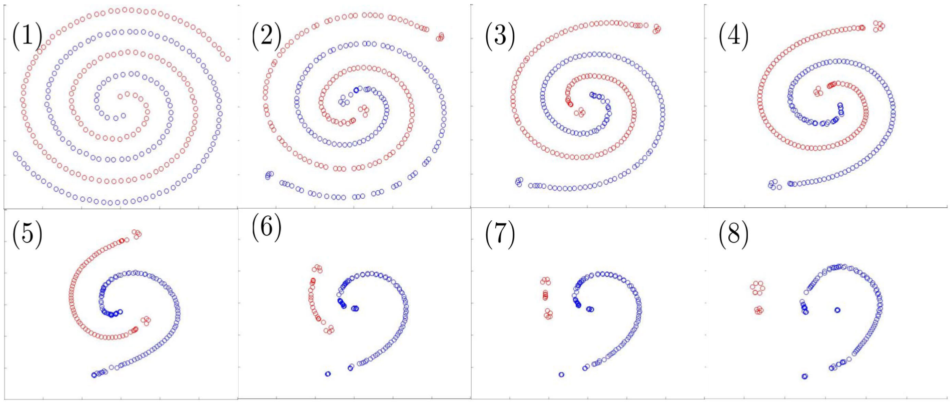

Figure 6.3: [103, Fig. 10], $F_n(x_i)$ for 8 different values of n.

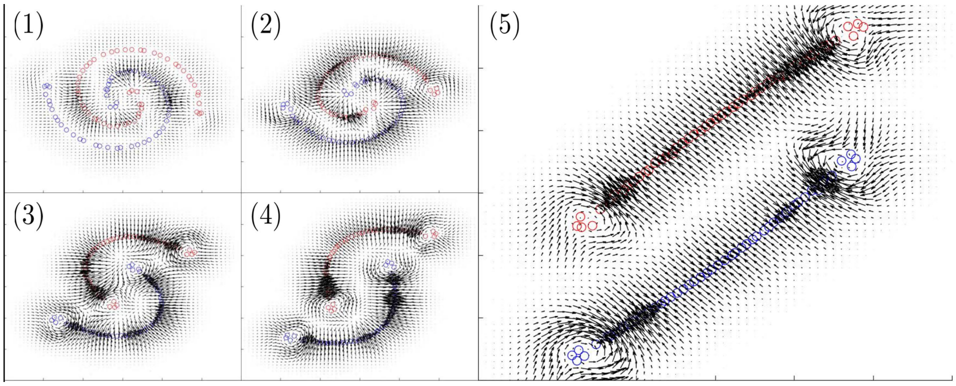

Figure 6.4: [103, Fig. 13], $(F_n(x_i))_{1\le i\le N}$ (dots) and $(F_{n+300}(x)-F_n(x))/300$ (arrows) for 5 different values of n.

[4] i.e., $F_n(x_i)$, where F_n is the deformation F at the n-th iteration

The MNIST and Fashion-MNIST databases

We will now implement, test and analyze the non-parametric KF algorithm applied to the MNIST dataset [83] as originally presented in [103, Sec. 8,9]. This training set is composed of 60000, 28×28 images of handwritten digits (partitioned into 10 classes) with a corresponding vector of 60000 labels (with values in $\{1, \ldots, 9, 0\}$). The test set is composed of 10000, 28×28 images of handwritten digits with a corresponding vector of 10000 labels. The Fashion-MNIST set [144] has identical data structure, though consists of images of articles of clothing of the 10 classes: T-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. We will highlight the following observations:

1. the KF algorithm applies class specific distortions, termed *class archetypes*.

2. the interpolation with the learned kernel from the KF algorithm is accurate when using a subset of the training data, even for subsets containing 10 total images with one representative per class.

3. the KF flow clusters images of like-class and separates images of differing class.

4. in the class of sandals, the KF algorithm clusters this single class into two sub-class, those with high heels and flat bottoms, which provides evidence of unsupervised learning.

N_I	Average error	Min error	Max error	Standard Deviation
6000	0.014	0.0136	0.0143	1.44×10^{-4}
600	0.014	0.0137	0.0142	9.79×10^{-5}
60	0.0141	0.0136	0.0146	2.03×10^{-4}
10	0.015	0.0136	0.0177	7.13×10^{-4}

Table 6.1: MNIST test errors using N_I interpolation points

In these experiments, we run 12000 iterations using batch sizes $N_f = 600$, $N_c = 300$ and initial kernel $k(x, x') = \exp(-\gamma \|x - x'\|^2)$. In Table 6.1, we present the error statistics[5] of using k_F to interpolate a random subset of N_I training points. Note that for $N_I = 10$ and $N_I = 60$, we require the random subset include one or six

[5] Error measured between steps 11901 and 12000 using random training subsets.

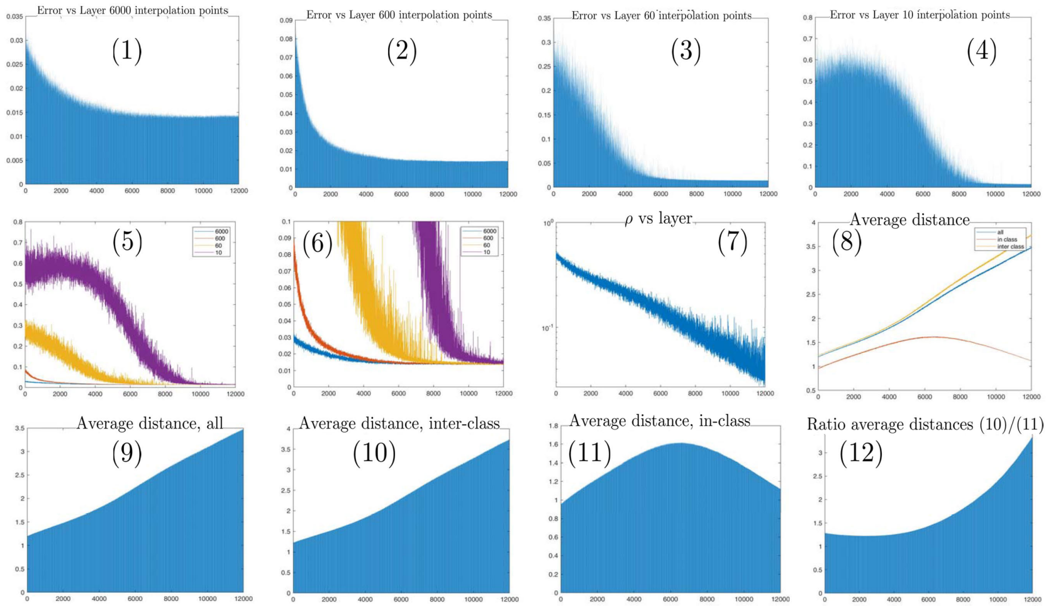

Figure 6.5: [103, Fig. 14], results for MNIST. $N = 60000$, $N_f = 600$ and $N_c = 300$. (1) Test error vs depth n with $N_I = 6000$ (2) Test error vs depth n with $N_I = 600$ (3) Test error vs depth n with $N_I = 60$ (4) Test error vs depth n with $N_I = 10$ (5,6) Test error vs depth n with $N_I = 6000, 600, 60, 10$ (7) ρ vs depth n (8) Mean-squared distances between images $F_n(x_i)$ (all, inter class and in class) vs depth n (9) Mean-squared distances between images (all) vs depth n (10) Mean-squared distances between images (inter class) vs depth n (11) Mean-squared distances between images (in class) vs depth n (12) Ratio $(10)/(11)$.

representatives from each class, respectively. We observe relatively stable error rates when reducing the interpolation size N_I. Even an interpolation with a random single image per class yields similar results to one with $N_I = 6000$ points.

Figure 6.5 shows test errors vs depth n (with $N_I = 6000, 600, 60, 10$ interpolation points), the value of the ratio ρ vs n (computed with $N_f = 600$ and $N_c = 300$) and the mean squared distances between (all, inter class and in class) images $F_n(x_i)$ vs n. Observe that all mean-squared distances increase until $n \approx 7000$. After $n \approx 7000$ the in class mean-squared distances decreases with n whereas the inter-class mean-squared distances continue increasing. This suggests that after $n \approx 7000$ the algorithm starts clustering the data per class. Note also that while the test errors, with $N_I = 6000, 600$ interpolation points, decrease immediately and sharply, the test errors with $N_I = 10$ interpolation points increase slightly until $n \approx 3000$ towards 60%, after which they drop and seem to stabilize around 1.5% towards $n \approx 10000$.

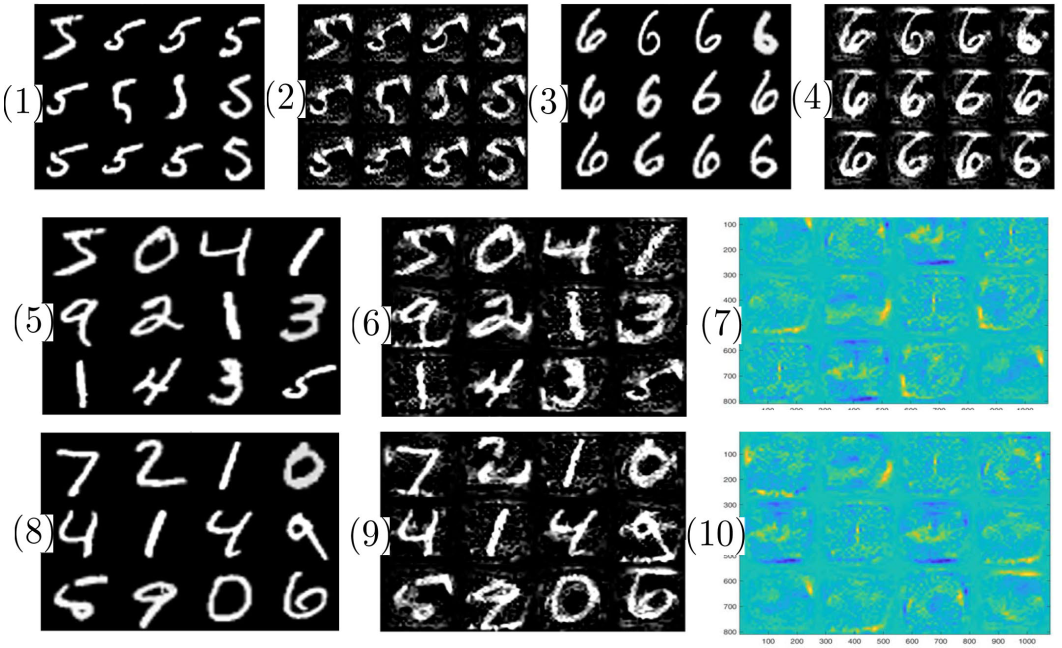

Figure 6.6: [103, Fig. 15], results for MNIST. $N = 60000$, $N_f = 600$ and $N_c = 300$. (1, 3, 5) Training data x_i (2, 4, 6) $F_n(x_i)$ for $n = 12000$ (7) $F_n(x_i) - x_i$ for training data and $n = 12000$ (8) Test data x_i (9) $F_n(x_i)$ for test data and $n = 12000$ (10) $F_n(x_i) - x_i$ for test data and $n = 12000$.

It is known that iterated random functions typically converge because they are contractive on average [31, 42]. Here training appears to create iterated functions that are contractive with each class but expansive between classes.

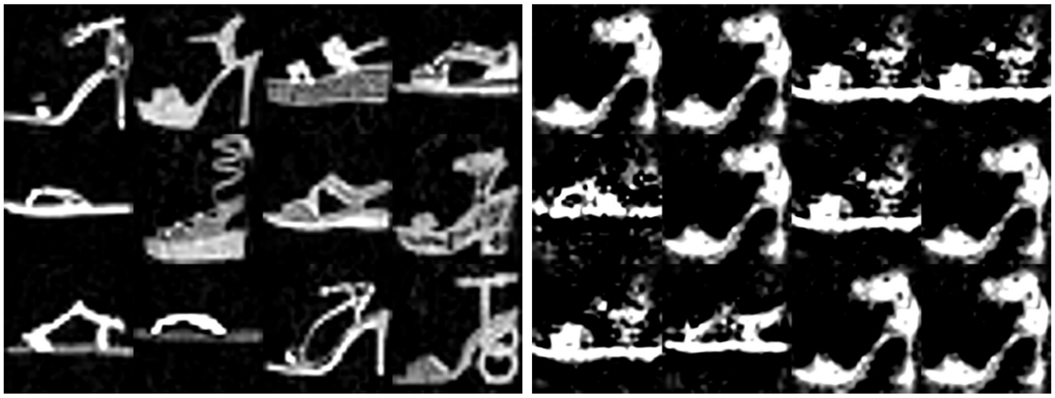

Figure 6.7: [103, Fig. 22], results for Fashion-MNIST. $N = 60000$, $N_f = 600$ and $N_c = 300$. Left: Training data x_i for class 5. Right: $F_n(x_i)$ training data and $n = 11000$.

Figure 6.6.1-4 shows the KF flow on 12 representatives of images of 5 and 6. We observe that a short vertical line is added to the upper right corner of 5's and a longer horizontal line is added to the top of each 6. Further observing in Figure 6.6.5-10, the KF algorithm appears to introduce small, archetypal, and class dependent, perturbations in those images.

A sign of unsupervised learning? Figure 6.7 shows x_i and $F_n(x_i)$ for a group of images in the class 5 (sandal). The network is trained to depth $n = 11000$. Surprisingly, the flow F_n accurately clusters that class (sandal) into 2 sub-classes: (1) high heels (2) flat bottom. This is surprising because the training labels contain no information about such sub-classes: the KF algorithm has separated those clusters/sub-classes without supervision.

Chapter 7

KERNEL FLOWS REGULARIZED NEURAL NETWORKS

We introduce a novel technique for regularizing artificial neural networks (ANNs) using the l_2-norm loss function from the Kernel Flow (KF) algorithm [154] as summarized in section 6.1. Conventional methods of training involve optimizing a loss function dependent on the final output of the ANN. In our method, we construct loss function to be the weighted sum of conventional and KF loss functions with kernel dependent on the inner layer outputs of the ANN. In this way, optimizing our loss function is a novel technique for simultaneously training the outputs of multiple layers of the ANN. We test the proposed kernel method on Convolutional Neural Networks (CNNs) and Wide Residual Networks (WRNs) without alteration of their structure nor their output classifier. With the incorporation of KF loss, we report reduced test errors, decreased generalization gaps, and increased robustness to distribution shift without significant increase in computational complexity relative to standard CNN and WRN training.

We proceed towards defining our KF loss in the context of a general ANN. Begin by writing

$$f_\theta(x) = \big(f^{(n)}_{\theta_n} \circ f^{(n-1)}_{\theta_{n-1}} \circ \cdots \circ f^{(1)}_{\theta_1}\big)(x) \tag{7.0.1}$$

for the compositional structure of an ANN with input x and n layers $f^{(i)}_{\theta_i}(z) = \phi(W_i z + b_i)$ parameterized by the weights and biases $\theta_i := (W_i, b_i), \theta := \{\theta_1, \dots, \theta_n\}$. The output of the ANN, $f_\theta(x)$, lies in domain $\mathbb{R}^{n_{\text{cl}}}$, where n_{cl} is the number of classes in the classification problem. We will use ReLU for the non-linearity ϕ in our experiments. For $i \in \{1, \dots, n-1\}$ let $h^{(i)}_\theta(x)$ be the output of the i-th (inner) layer, i.e.

$$h^{(i)}_\theta(x) := \big(f^{(i)}_{\theta_i} \circ f^{(i-1)}_{\theta_{i-1}} \circ \cdots \circ f^{(1)}_{\theta_1}\big)(x)\,, \tag{7.0.2}$$

and let $h_\theta(x) := (h^{(1)}_\theta(x), \dots, h^{(n-1)}_\theta(x))$ be the $(n-1)$-ordered tuple representing all inner layer outputs. Let $k_\gamma(\cdot,\cdot)$ be a family of kernels parameterized by γ and let $K_{\gamma,\theta}$ be the family of kernels parameterized by γ and θ defined by

$$K_{\gamma,\theta}(x, x') = k_\gamma(h_\theta(x), h_\theta(x'))\,. \tag{7.0.3}$$

Define

$$\mathcal{L}_{\text{c-e}}(f_\theta(x), y) := \sum_{j=1}^{n_{\text{cl}}} y_j \log \big(f_\theta(x)\big)_j \tag{7.0.4}$$

and given the random mini-batch $(\mathbf{X^f}, \mathbf{Y^f})$

$$\mathcal{L}_{\text{c-e}}(f_\theta(\mathbf{X^f}), \mathbf{Y^f}) := \sum_i \mathcal{L}_{\text{c-e}}(f_\theta(X^f_i), Y^f_i) \tag{7.0.5}$$

be the cross-entropy loss[1] associated with that mini-batch. Given the (randomly subsampled) half sub-batch (X^c, Y^c), let $\mathcal{L}_{\text{KF}}(\gamma, \theta, \mathbf{X^f}, \mathbf{Y^f}, \mathbf{X^c}, \mathbf{Y^c})$ be the loss function (with hyper-parameter $\lambda \geq 0$) defined by

$$\mathcal{L}_{\text{KF}} := \lambda \|\mathbf{Y^f} - \mathbf{K}_{\gamma,\theta}(\mathbf{X^f}, \mathbf{X^c})\mathbf{K}_{\gamma,\theta}(\mathbf{X^c}, \mathbf{X^c})^{-1}\mathbf{Y^c}\|_2^2 + \mathcal{L}_{\text{c-e}}(f_\theta(X^f), Y^f)\,. \tag{7.0.6}$$

Our proposed KF-regularization approach is then to train the parameters θ of the network f_θ via the steepest descent $(\gamma, \theta) \leftarrow (\gamma, \theta) - \delta \nabla_{\gamma,\theta} \mathcal{L}_{\text{KF}}$. Network training with gradient descent must be initialized with θ_0, γ_0 and specify learning rate δ, which is usually taken to be exponentially decreasing. Note that this algorithm

1. is randomized through both the sampling of the minibatch and its subsampling.
2. adapts both θ and γ (since the KF loss term depends on both θ and γ).
3. simultaneously trains the accuracy of the output via the cross-entropy term and the generalization properties of the feature maps defined by the inner layers via the KF loss term.

Furthermore while the cross-entropy term is a linear functional of the empirical distribution $\frac{1}{N_b}\sum_i \delta_{(X^f_i, Y^f_i)}$ defined by the mini-batch (writing N_b for the number of indices contained in the mini-batch), the KF loss function is non-linear. While $K_{\gamma,\theta}$ may depend on the output of all the inner layers, in our numerical experiments we have restricted its dependence to the output of only one inner layer or used a weighted sum of such terms.

7.1 Numerical experiments

We will now use the proposed KF-regularization method to train a simple Convolutional Neural Network (CNN) on MNIST and Wide Residual Networks (WRN) [155] on fashion MNIST, CIFAR-10, and CIFAR-100. These results are also presented in

[1] This is the loss function most commonly utilized by ANN training.

[154, Sec. 3]. The conventional approach for training such networks is optimizing the cross-entropy loss function while using Batch Normalization (BN) and Drop Out (DO). Summarizing these techniques, BN [69] normalizes the distribution of every hidden layer output of each training batch. In doing so, BN allows for higher learning rates and being less careful about network initialization, implying it adds stability to the training of the network. Furthermore, DO [125] randomly removes components within each linear mapping layer of the network when training. It is known to reduce overfitting of data and improve performance of networks. Our goal is to test our proposed kernel approach and compare its performance with these conventional training techniques.

Kernel Flow regularization on MNIST

We consider a Convolutional Neural Network (CNN) with six convolutional layers and three fully connected layers, as charted in Table 7.1 (this CNN is a variant of a CNN presented in [28] with code used from [56]). Convolutional layers consist of an image convolution with a multiple filters. All layers in this network have stride one, meaning the convolution is evaluated at every pixel[2]. The size of the convolutional kernel is shown in the second and third columns from the left. "Valid" padding implies no 0-padding at the boundaries of the image while "same" 0-pads images to obtain convolutional outputs with the same sizes as the inputs. The "Max Pool" layers down sample their inputs by reducing each 2×2 square to their maximum values. The "Average Pool" layer in the final convolutional layer takes a simple mean over each channel. The final three layers are fully connected, or equivalently dense, each with outputs listed on the right column. Fully connected layers are arbitrary linear maps, meaning dense layer 1 corresponds to an $\mathbb{R}^{300\times 1200}$ matrix. All convolutional and dense layers include trainable biases. Using notations from the previous section, the outputs of the convolutional layers, which include ReLU and pooling, are $h^{(1)}(x)$ to $h^{(6)}(x)$ with output shapes described in the left column. The dense layers outputs are $h^{(7)}(x)$ to $h^{(9)}(x)$. We do not pre-process the data and, when employed, the data augmentation step, in this context, passes the original MNIST image to the network with probability $\frac{1}{3}$, applies an elastic deformation [122] with probability $\frac{1}{3}$, and a random small translation, rotation, and shear with probability $\frac{1}{3}$. The learning rate, as selected by validation, begins at 10^{-3} and smoothly exponentially decreases to 10^{-7} while training over 20 epochs.

[2] A stride of k implies the convolution is evaluated every k pixels.

Layer Type	Number of filters	Filter size	Padding	Output shape
Input layer				$28 \times 28 \times 1$
Convolutional layer 1, ReLU	150	3×3	Valid	$26 \times 26 \times 150$
Convolutional layer 2, ReLU	150	3×3	Valid	$24 \times 24 \times 150$
Convolutional layer 3, ReLU	150	5×5	Same	$24 \times 24 \times 150$
Max Pool		2×2		$12 \times 12 \times 150$
Convolutional layer 4, ReLU	300	3×3	Valid	$10 \times 10 \times 300$
Convolutional layer 5, ReLU	300	3×3	Valid	$8 \times 8 \times 300$
Convolutional layer 6, ReLU	300	5×5	Same	$8 \times 8 \times 300$
Max Pool		2×2		$4 \times 4 \times 300$
Average Pool		4×4		300
Dense layer 1, ReLU				1200
Dense layer 2, ReLU				300
Dense layer 3				10
Softmax Output layer				10

Table 7.1: The architecture of the CNN used in KF-regularization experiments is charted. Convolutional layers are divided with horizontal lines. The middle block shows layer specifics and the shapes of the outputs of each layer is on the right.

Comparisons of dropout and KF-regularization

The first experiment we present results of training the CNN with architecture given in Table 7.1 with (1) Batch Normalization (BN) [69] (2) BN and KF-regularization (3) BN and dropout (DO) [125] (4) BN, KF-regularization, and DO. We use the same dropout structure as in [28], and use a rate of 0.3, as selected with validation.

Training Method	Original MNIST	Data augmented	QMNIST
BN only	0.395 ± 0.030%	0.302 ± 0.026%	0.389 ± 0.014%
BN+KF	0.300 ± 0.024%	0.281 ± 0.033%	0.341 ± 0.013%
BN+DO	0.363 ± 0.028%	0.314 ± 0.024%	0.400 ± 0.015%
BN+KF+DO	0.296 ± 0.023%	0.287 ± 0.022%	0.344 ± 0.015%

Table 7.2: A comparison of the average and standard deviation of testing errors each over 20 runs for networks. The first data column on the left shows networks trained and tested on original MNIST data. The middle is trained using data augmentation and uses original MNIST testing data. The right column shows the same data augmented trained network, but uses QMNIST testing data [149].

We present a KF-regularization experiment using the following Gaussian kernel on the final convolutional layer $h^{(6)}(x) \in \mathbb{R}^{300}$:

$$\begin{aligned} K^{(6)}_{\gamma_6,\theta}(x,x') &= k^{(6)}_{\gamma_6}(h^{(6)}(x), h^{(6)}(x')) \\ &= e^{-\gamma_6 \|h^{(6)}(x)-h^{(6)}(x')\|^2}. \end{aligned} \tag{7.1.1}$$

We optimize the loss function in (7.0.6) with kernel $K^{(6)}_{\gamma_6}$ over the parameters θ and γ_6. Specifically, given the random mini-batch (X^b, Y^b) and the (randomly sub-sampled) half sub-batch (X^c, Y^c), we evolve θ and γ_6 in the steepest descent direction of the loss

$$\begin{aligned}\mathcal{L}_{\text{KF}} &= \lambda^{(6)} \|Y^b - K^{(6)}_{\gamma_6,\theta}(X^b, X^c) K^{(6)}_{\gamma_6,\theta}(X^c, X^c)^{-1} Y^c\|_2^2 \\ &+ \mathcal{L}_{\text{c-e}}(f_\theta(X^b), Y^b)\,.\end{aligned} \tag{7.1.2}$$

The comparison between the dropout and KF-regularization training methods, as well as their combination, is made in Table 7.2. KF-regularization and the network architecture was inspired by the work in [103, Sec. 10] (the GPR estimator on the final convolutional output space is here replaced by a fully connected network to minimize computational complexity). On a 12GB NVIDIA GeForce GTX TITAN X graphics card, training one network with BN+DO (20 epochs) takes 1605s to run, compared with 1629s for BN+KF+DO. Furthermore, this KF-regularization framework has another advantage of being flexible, both allowing the control of generalization properties of multiple layers of the network simultaneously and being able to be used concurrently with dropout.

For each of the training methods, we experiment with using original MNIST training and testing data, augmenting the MNIST training set and testing on the original data, and finally training on the augmented set, but testing on QMNIST, which is resampled MNIST test data [149]. These three regimes are presented in the data columns of Table 7.2 from left to right. The difference between the original data augmented and QMNIST testing errors quantifies the effect of distributional shift of the testing data [111]. This effect is observed to be reduced when using KF-regularized trained networks, which suggests some degree of robustness to distributional shift.

The training and testing errors of single runs of networks trained with BN only, BN+DO, BN+KF, and BN+KF+DO are plotted in Fig. 7.1. Observe that the generalization gap (the gap between the training and testing errors) decreases with the use of dropout, and that decrease is even more pronounced in the experiments with KF-regularization. We observe similar findings on networks trained using data augmentation, albeit less pronounced. We finally examine the KF-regularization component of the loss function as in equation (7.1.2). This KF loss, $\|Y^b - K^{(6)}_{\gamma_6,\theta}(X^b, X^c) K^{(6)}_{\gamma_6,\theta}(X^c, X^c)^{-1} Y^c\|_2^2$, is computed for batch normalization, dropout, and KF-regularized training in Fig. 7.2. Both BN+KF and BN+KF+DO

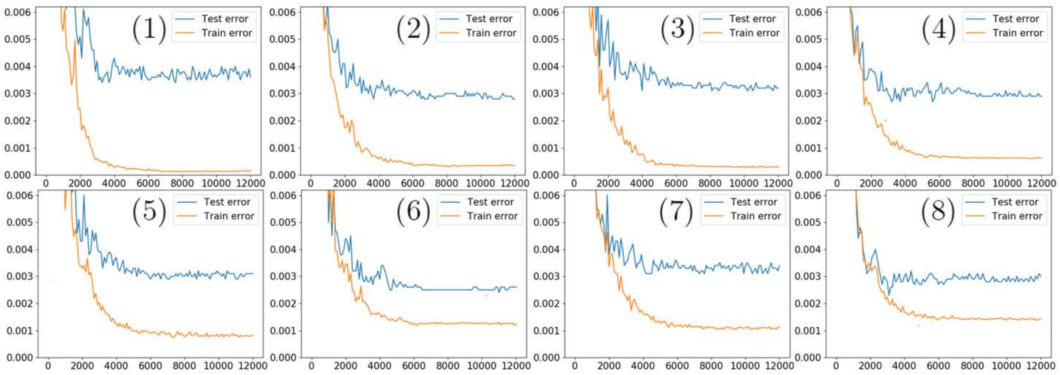

Figure 7.1: [154, Fig. 2], training and testing errors are plotted over single runs trained with original data using (1) BN only (2) BN+KF (3) BN+DO (4) BN+KF+DO. Data augmented trained network errors are shown using (5) BN only (6) BN+KF (7) BN+DO (8) BN+KF+DO.

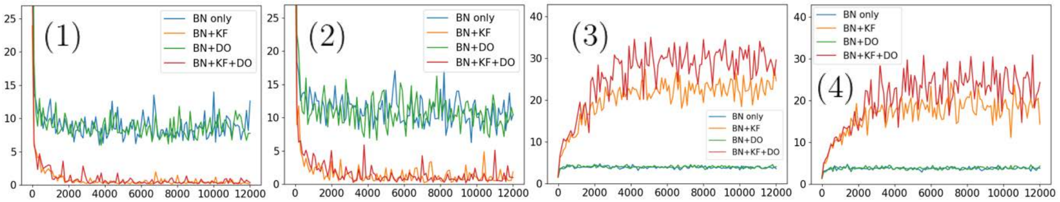

Figure 7.2: [103, Fig. 3], single run with each of BN only, BN+KF, BN+DO, and BN+KF+DO training methods plotting (1) 6th layer KF-loss using the original MNIST training set (2) 6th layer KF-loss using an augmented training set (3) ratio of mean inter-class and in-class distances of 6th layer outputs using the original training set (4) ratio of mean inter-class and in-class distances of 6th layer outputs using an augmented set.

are observed to reduce the KF loss and increase the ratio of inter-class and in-class pairwise distances within each batch. The class-dependent clustering within hidden layer outputs highlights the difference between traditional training techniques and KF-regularization.

Kernel Flow regularization on Fashion MNIST and CIFAR

We now consider the Wide Residual Network (WRN) structure described in [155, Table 1] with the addition of a dense layer. For convenience, we show this architecture in Table 7.3. Note that there are four convolutional blocks, each with a certain number of residual layers, which are as described in [155, Fig. 1c,d] for

Layer/Block name	Number of filters	Filter size	Number of residual layers	Output shape
Input layer				$32 \times 32 \times 3$
Convolutional block 1	16	3×3	1	$32 \times 32 \times 16$
Convolutional block 2	$16k$	3×3	N	$32 \times 32 \times 16k$
Convolutional block 3	$32k$	3×3	N	
Max Pool		2×2		$16 \times 16 \times 32k$
Convolutional block 4	$64k$	3×3	N	
Max Pool		2×2		$8 \times 8 \times 64k$
Average Pool		8×8		$64k$
Dense layer				$64k$
Softmax Output layer				10

Table 7.3: The architecture of the WRN used in KF-regularization experiments with CIFAR input images. Convolutional blocks are divided with horizontal lines. The middle portion shows block specifics such as filter width and depth in each block and the shapes of the outputs of each layer is on the right. Note that max pooling occurs within the last residual layer of each block.

BN and BN+DO training respectively. Each layer consists of two convolutional blocks, with dropout applied between the blocks in dropout training, added to an identity mapping from the input of the layer. In our dropout experiments, we drop each neuron in the network with probability 0.3, as selected with cross-validation in [155]. Note that k and N are hyper-parameters of the WRN architecture governing width and depth, respectively, and a network with such k, N is written WRN-k-N. In these presented WRN experiments, we use data augmentation where training images are randomly translated and horizontally flipped. In our implementations, we have modified the code from [73] (which uses TensorFlow). Batches consisting of 100 images are used in these experiments. In CIFAR-10, each half batch contains 5 random images from each of the 10 classes. Meanwhile, in CIFAR-100, we require each class represented in the testing sub-batch to also be represented in the training sub-batch.

We write the outputs of each of the four convolutional blocks as $h^{(1)}(x), \ldots, h^{(4)}(x)$. Again defining a as the average pooling operator, we have $a(h^{(1)}(x)) \in \mathbb{R}^{16}$, $a(h^{(2)}(x)) \in \mathbb{R}^{16k}$, $a(h^{(3)}(x)) \in \mathbb{R}^{32k}$, and $a(h^{(4)}(x)) = h^{(4)}(x) \in \mathbb{R}^{64k}$. We define corresponding RBF kernels

$$\begin{aligned} K_{\gamma_l}^{(l)}(x, x') &= k_{\gamma_l}^{(l)}(h^{(l)}(x), h^{(l)}(x')) \\ &= e^{-\gamma_l \|a(h^{(l)}(x)) - a(h^{(l)}(x'))\|^2}. \end{aligned} \tag{7.1.3}$$

Given the random mini-batch (X^b, Y^b) and the (randomly sub-sampled) half sub-batch (X^c, Y^c), we evolve θ (and γ) in the steepest descent direction of the loss

$$\begin{aligned} \mathcal{L}_{\text{KF}} = \sum_{l=1}^{4} \lambda^{(l)} \| Y^b - K^{(l)}_{\gamma_l,\theta}(X^b, X^c) K^{(l)}_{\gamma_l,\theta}(X^c, X^c)^{-1} Y^c \|_2^2 \\ + \mathcal{L}_{\text{c-e}}(f_\theta(X^b), Y^b) \, . \end{aligned} \tag{7.1.4}$$

Comparison to Dropout

Training Method	CIFAR-10	CIFAR-10.1	CIFAR-100
BN	4.72 ± 0.17%	11.07 ± 0.55%	20.42 ± 0.25%
BN+KF	4.43 ± 0.12%	10.38 ± 0.40%	20.37 ± 0.27%
BN+DO	4.39 ± 0.08%	10.50 ± 0.39%	19.58 ± 0.41%
BN+KF+DO	4.05 ± 0.11%	10.20 ± 0.32%	19.38 ± 0.18%

Table 7.4: A comparison of the average and standard deviation of test errors over 5 runs for networks trained on augmented data on CIFAR-10, CIFAR-10.1, and CIFAR-100. The second column to the right trains on augmented CIFAR-10 data but tests on CIFAR-10.1 data [110, 135]..

Table 7.4 compares the test errors obtained after training with only batch normalization (BN) with the incorporation of dropout (DO), KF-regularization, as well as a combination of all three. The network architecture WRN-16-8 is used and testing error statistics over five runs is listed. We train with step exponentially decreasing learning rates over 200 epochs with identical hyperparameters as [155]. We observe that KF-regularization improves testing error rates against training with BN and BN+DO. We also run a distributional shift experiment for CIFAR-10 using the data set CIFAR-10.1, [110] which is sampled from [135]. As with the QMNIST experiment, we also observe improvements with the addition of KF-regularization.

We finally compare the KF loss, $\mathcal{L}_{\text{KF}}$, and ratios of inter-class and in-class Euclidean distances on the output of the final convolutional layers within each batch in Figure 7.3. These statistics are plotted over runs of WRN trained with CIFAR-10 and CIFAR-100. We again observe reduced KF losses and increased ratios of mean inter-class and in-class distances on the final convolutional layer output $h^{(4)}$ when comparing between BN and BN+KF as well as BN+DO and BN+KF+DO. That is, KF-regularization reduces the distance (defined on the outputs of the inner layers) between images in the same class and increases the distance between images in

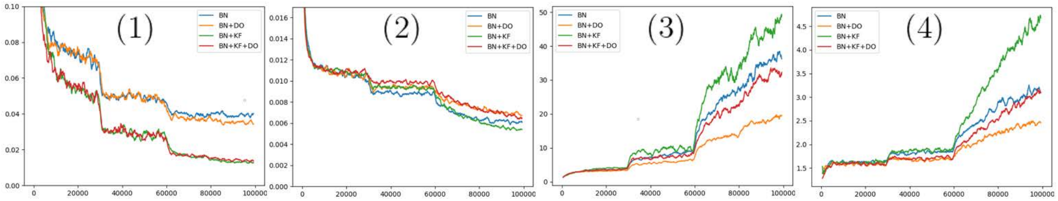

Figure 7.3: [154, Fig. 4], single run using WRN-16-8 with each of BN only, BN+KF, BN+DO, and BN+KF+DO plotting (1) CIFAR-10 KF loss (2) CIFAR-100 KF loss (3) CIFAR-10 ratio of mean inter-class and in-class distances $h^{(4)}$ (4) CIFAR-100 ratio of mean inter-class and in-class distances $h^{(4)}$.

distinct classes (thereby enhancing the separation). The opposite effect is observed with the addition of dropout in training, suggesting they improve testing errors through distinct mechanisms.

BIBLIOGRAPHY

[1] F.E. Ahangar, F.R. Freedman, and A." Venkatram. Using low-cost air quality sensor networks to improve the spatial and temporal resolution of concentration maps. *International Journal of Environmental Research and Public Health*, 16(7):1252, 2019. https://doi.org/10.3390/ijerph16071252.

[2] R. Angus, T. Morton, S. Aigrain, D. Foreman-Mackey, and V. Rajpaul. Inferring probabilistic stellar rotation periods using Gaussian processes. *Monthly Notices of the Royal Astronomical Society*, 474(2):2094–2108, 2018. https://doi.org/10.1093/mnras/stx2109.

[3] F. Bachoc. Cross validation and maximum likelihood estimations of hyper-parameters of Gaussian processes with model misspecification. *Computational Statistics & Data Analysis*, 66:55–69, 2013. https://doi.org/10.1016/j.csda.2013.03.016.

[4] F. Bachoc. Asymptotic analysis of covariance parameter estimation for Gaussian processes in the misspecified case. *Bernoulli*, 24:1531–1575, 2018. https://doi.org/10.3150/16-BEJ906.

[5] F. Bachoc, A. Lagnoux, and A.F. López-Lopera. Maximum likelihood estimation for Gaussian processes under inequality constraints. *Electronic Journal of Statistics*, 13(2):2921–2969, 2019. https://doi.org/10.1214/19-EJS1587.

[6] F. Bachoc, A. Lagnoux, and T.M.N. Nguyen. Cross-validation estimation of covariance parameters under fixed-domain asymptotics. *Journal of Multivariate Analysis*, 160:42–67, Aug. 2017. https://doi.org/10.1016/j.jmva.2017.06.003.

[7] N.A. Baker, D. Sept, S. Joseph, M.J. Holst, and J.A. McCammon. Electrostatics of nanosystems: application to microtubules and the ribosome. *Proceedings of the National Academy of Sciences*, 98(18):10037–10041, 2001.

[8] J.C.A. Barata and M.S. Hussein. The Moore–Penrose pseudoinverse: A tutorial review of the theory. *Brazilian Journal of Physics*, 42(1-2):146–165, 2012. https://doi.org/10.1007/s13538-011-0052-z.

[9] B.M. Battistia, C. Knapp, K. McGee, and V. Goebel. Application of the empirical mode decomposition and Hilbert-Huang transform to seismic reflection data. *Geophysics*, 72, 2007. https://doi.org/10.1190/1.2437700.

[10] H. Bayraktar and F.S. Turalioglu. A Kriging-based approach for locating a sampling site—in the assessment of air quality. *Stochastic Environmental*

Research and Risk Assessment, 19(4):301–305, 2005. https://doi.org/10.1007/s00477-005-0234-8.

[11] R. Behera, S. Meignen, and T. Oberlin. Theoretical analysis of the second-order synchrosqueezing transform. *Applied and Computational Harmonic Analysis*, 45, 2018. https://doi.org/10.1016/j.acha.2016.11.001.

[12] S. Bellstedt, D.A. Forbes, C. Foster, A.J. Romanowsky, J.P. Brodie, N. Pastorello, A. Alabi, and A. Villaume. The SLUGGS survey: using extended stellar kinematics to disentangle the formation histories of low-mass S0 galaxies. *Monthly Notices of the Royal Astronomical Society*, 467(4):4540–4557, 2017. https://doi.org/10.1093/mnras/stx418.

[13] J. Braun and M. Griebel. On a constructive proof of Kolmogorov's superposition theorem. *Constructive Approximation*, 30(3):653, 2009. https://doi.org/10.1007/s00365-009-9054-2.

[14] K.-M. Chang. Arrhythmia ECG noise reduction by ensemble empirical mode decomposition. *Sensors*, 10(6):6063–6080, 2010. https://doi.org/10.3390/s100606063.

[15] J. Chen, B. Heincke, M. Jegen, and M. Moorkamp. Using empirical mode decomposition to process marinemagnetotelluric data. *Geophysical Journal International*, 190(1):293–309, 2012. https://doi.org/10.1111/j.1365-246X.2012.05470.x.

[16] J.-P. Chilès and N. Desassis. Fifty years of Kriging. In *Handbook of Mathematical Geosciences*, pages 589–612. Springer, Cham, 2018. https://doi.org/10.1007/978-3-319-78999-6_29.

[17] J.P. Chilès and P. Delfiner. *Geostatistics: Modeling Spatial Uncertainty*, volume 497. John Wiley & Sons, 2009.

[18] A. Chouldechova and T. Hastie. Generalized additive model selection. *arXiv preprint arXiv:1506.03850*, 2015.

[19] K.A. Cliffe, M.B. Giles, R. Scheichl, and A.L. Teckentrup. Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients. *Computing and Visualization in Science*, 14(1):3, 2011.

[20] J. Cockayne, C.J. Oates, T.J. Sullivan, and M. Girolami. Bayesian probabilistic numerical methods. *SIAM Review*, 61:756–789, 2019. https://doi.org/10.1137/17M1139357.

[21] J.A. Costa and A.O. Hero. Learning intrinsic dimension and intrinsic entropy of high-dimensional datasets. In *2004 12th European Signal Processing Conference*, pages 369–372. IEEE, 2004.

[22] K.T. Coughlin and K.-K. Tung. 11-year solar cycle in the stratosphere extracted by the empirical mode decomposition method. *Advances in Space Research*, 34:323–329, 2004. https://doi.org/10.1016/j.asr.2003.02.045.

[23] D.A.T. Cummings, R.A. Irizarry, N.E. Huang, T.P. Endy, A. Nisalak, K. Ungchusak, and D.S. Burke. Travelling waves in the occurrence of dengue haemorrhagic fever in Thailand. *Nature*, 427:344–347, 2004. https://doi.org/10.1038/nature02225.

[24] I. Daubechies, J. Lu, and H.-T. Wu. Synchrosqueezed wavelet transforms: an empirical mode decomposition-like tool. *Applied and Computational Harmonic Analysis*, 30:243–261, 2011. https://doi.org/10.1016/j.acha.2010.08.002.

[25] I. Daubechies, Y.G. Wang, and H.-T. Wu. ConceFT: concentration of frequency and time via a multitapered synchrosqueezed transform. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374, 2016. https://doi.org/10.1098/rsta.2015.0193.

[26] I.A. Delbridge, D.S. Bindel, and A.G. Wilson. Randomly projected additive Gaussian processes for regression. *arXiv preprint arXiv:1912.12834*, 2019.

[27] E. Deléchelle, J. Lemoine, and O. Niang. Empirical mode decomposition: an analytical approach for sifting process. *IEEE Signal Processing Letters*, 12(11):764–767, 2005. https://doi.org/10.1109/LSP.2005.856878.

[28] C. Deotte. *How to choose CNN Architecture MNIST*, 2019. https://www.kaggle.com/cdeotte/how-to-choose-cnn-architecture-mnist (accessed January, 2020).

[29] A.U. Dey and G. Harit. Gradient sensitive kernel for Image Denoising, using Gaussian Process Regression. In *2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*. IEEE, 2015. https://doi.org/10.1109/NCVPRIPG.2015.7490043.

[30] P. Diaconis. Bayesian numerical analysis. In *Statistical Decision Theory and Related Topics, IV, Vol. 1 (West Lafayette, Ind., 1986)*, pages 163–175. Springer, New York, 1988.

[31] P. Diaconis and D. Freedman. Iterated random functions. *SIAM review*, 41(1):45–76, 1999. https://doi.org/10.1137/S0036144598338446.

[32] J. Dinarés-Ferran, R. Ortner, C. Guger, and J. Solé-Casals. A new method to generate artificial frames using the empirical mode decomposition for an EEG-based motor imagery BCI. *Frontiers in Neuroscience*, 12, 2018. https://doi.org/10.3389/fnins.2018.00308.

[33] L. Ding and P. Mathé. Minimax rates for statistical inverse problems under general source conditions. *Computational Methods in Applied Mathematics*, 18(4):603–608, 2018. https://doi.org/10.1515/cmam-2017-0055.

[34] R. Djemili, H. Bourouba, and M.C. Amara Korba. Application of empirical mode decomposition and artificial neural network for the classification of normal and epileptic EEG signals. *Biocybernetics and Biomedical Engineering*, 36:285–291, 2016. https://doi.org/10.1016/j.bbe.2015.10.006.

[35] D.-Q. Dong, J.-Y. Wang, Tao An, H.-B. Qiu, and X.-L. Lu. Multiple periodic oscillations analysis on astronomy signals using an Empirical Mode Decomposition-Ornstein Uhlenbeck method. In *2015 8th International Congress on Image and Signal Processing (CISP)*, pages 1178–1183. IEEE, 2015. https://doi.org/10.1109/CISP.2015.7408059.

[36] D. Donoho and I Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994. https://doi.org/10.1093/biomet/81.3.425.

[37] D.L. Donoho. De-noising by soft-thresholding. *IEEE transactions on information theory*, 41(3):613–627, 1995. https://doi.org/10.1109/18.382009.

[38] D.L. Donoho and I.M. Johnstone. Minimax estimation via wavelet shrinkage. *The annals of Statistics*, 26(3):879–921, 1998. https://doi.org/10.1214/aos/1024691081.

[39] D.L. Donoho, R.C. Liu, and B. MacGibbon. Minimax risk over hyperrectangles, and implications. *The Annals of Statistics*, pages 1416–1437, 1990. https://www.jstor.org/stable/2242061.

[40] K. Dragomiretskiy and D. Zosso. Variational mode decomposition. *IEEE Transactions on Signal Processing*, 62(3):531–544, 2013. https://doi.org/10.1109/TSP.2013.2288675.

[41] O. Dubrule. Comparing splines and kriging. *Computers & Geosciences*, 10:327–338, 1984. https://doi.org/10.1016/0098-3004(84)90030-X.

[42] M.M. Dunlop, M.A. Girolami, A.M. Stuart, and A.L. Teckentrup. How deep are deep Gaussian processes? *The Journal of Machine Learning Research*, 19(1):2100–2145, 2018. https://dl.acm.org/doi/abs/10.5555/3291125.3309616.

[43] N. Durrande, D. Ginsbourger, and O. Roustant. Additive covariance kernels for high-dimensional Gaussian process modeling. volume 21, pages 481–499, 2012. https://afst.centre-mersenne.org/item/AFST_2012_6_21_3_481_0/.

[44] N. Durrande, J. Hensman, M. Rattray, and N.D. Lawrence. Detecting periodicities with Gaussian processes. *PeerJ Computer Science*, 2, 2016. https://doi.org/10.7717/peerj-cs.50.

[45] D.K. Duvanaud, H. Nickisch, and C.E. Rasmussen. Additive Gaussian processes. In *Advances in neural information processing systems*, pages 226–234, 2011.

[46] D. Foreman-Mackey, E. Agol, S. Ambikasaran, and R. Angus. Fast and scalable Gaussian process modeling with applications to astronomical time series. *The Astronomical Journal*, 154(6), 2017. https://doi.org/10.3847/1538-3881/aa9332.

[47] L. Foster, A. Waagen, N. Aijaz, M. Hurley, A. Luis, J. Rinsky, C. Satyavolu, M.J. Way, P. Gazis, and A. Srivastava. Stable and efficient Gaussian process calculations. *Journal of Machine Learning Research*, 10:857–882, 2009.

[48] D. Gabor. Theory of communication. Part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, 93(26):429–441, 1946.

[49] S. Gaci. A new ensemble empirical mode decomposition (EEMD) denoising method for seismic signals. *Energy Procedia*, 97:84–91, 2016. https://doi.org/10.1016/j.egypro.2016.10.026.

[50] S. Gaci. Seismic signal denoising using empirical mode decomposition. In S Gaci and O. Hachay, editors, *Oil and Gas Exploration: Methods and Application*, chapter 3. Wiley, Hoboken, NJ, 2017. https://doi.org/10.1002/9781119227519.ch3.

[51] H. Ge, G. Chen, H. Yu, H. Chen, and F. An. Theoretical analysis of empirical mode decomposition. *Symmetry*, 10(11), 2018. https://doi.org/10.3390/sym10110623.

[52] E. Gilboa, Y. Saatci, and J.P. Cunningham. Scaling multidimensional Gaussian processes using projected additive approximations. In *International Conference on Machine Learning*, pages 454–461, 2013.

[53] D. Ginsbourger, D. Dupuy, A. Badea, L. Carraro, and O. Roustant. A note on the choice and the estimation of Kriging models for the analysis of deterministic computer experiments. *Applied Stochastic Models in Business and Industry*, 25(2):115–131, 2009. https://doi.org/10.1002/asmb.741.

[54] M. Gonen and E. Alpaydin. Multiple kernel learning algorithms. *The Journal of Machine Learning Research*, 12:2211–2268, 2011.

[55] J.L. Gómez and D.R. Velis. A simple method inspired by empirical mode decomposition for denoising seismic data. *Geophysics*, 81, 2016. https://doi.org/10.1190/geo2015-0566.1.

[56] M. Görner. *Tensorflow and deep learning without a PhD*, 2019. https://github.com/GoogleCloudPlatform/tensorflow-without-a-phd (accessed December, 2019).

[57] Boumediene Hamzi and Houman Owhadi. Learning dynamical systems from data: a simple cross-validation perspective. *arXiv preprint arXiv:2007.05074*, 2020.

[58] J. Han and M. van der Baan. Empirical mode decomposition for seismic time-frequency analysis. *Geophysics*, 78, 2013. https://doi.org/10.1190/geo2012-0199.1.

[59] T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, 1(3):297–310, 1986.

[60] S.D. Hawley, L.E. Atlas, and H.J. Chizeck. Some properties of an empirical mode type signal decomposition algorithm. *IEEE Signal Processing Letters*, 17(1):24–27, 2009. https://doi.org/10.1109/LSP.2009.2030855.

[61] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on Imagenet Classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[62] L. He, M. Lech, N.C. Maddage, and N.B. Allen. Study of empirical mode decomposition and spectral analysis for stress and emotion classification in natural speech. *Biomedical Signal Processing and Control*, 6:139–146, 2011. https://doi.org/10.1016/j.bspc.2010.11.001.

[63] R.H. Herrera, J. Han, and M. van der Baan. Applications of the synchrosqueezing transform in seismic time-frequency analysis. *Geophysics*, 79, 2014. https://doi.org/10.1190/geo2013-0204.1.

[64] T. Ho-Phuoc. CIFAR10 to compare visual recognition performance between deep neural networks and humans. *arXiv preprint arXiv:1811.07270*, 2018.

[65] T.Y. Hou and Z. Shi. Adaptive data analysis via sparse time-frequency representation. *Advances in Adaptive Data Analysis*, 3(1&2):1–28, 2011. https://doi.org/10.1142/S1793536911000647.

[66] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1971):903–995, 1998. https://doi.org/10.1098/rspa.1998.0193.

[67] N.E. Huang. *Hilbert-Huang Transform and its Applications*. Interdisciplinary Mathematical Sciences. World Scientific Publishing Company, 2nd edition, 2014.

[68] Y. Huang, H. Di, R. Malekian, X. Qi, and Z. Li. Noncontact measurement and detection of instantaneous seismic attributes based on complementary ensemble empirical mode decomposition. *Energies*, 10, 2017. https://doi.org/10.3390/en10101655.

[69] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[70] S. Jyothi, S.V.B. Rao, and P. Kishore. Natural periodic oscillations extracted in the precipitation using empirical mode decomposition and ensemble empirical mode decomposition methods. *International Journal of Current Research and Review*, 9, Oct 2017. http://doi.org/10.7324/IJCRR.2017.9192.

[71] S. Kampakis. *Predictive modelling of football injuries*. PhD thesis, University College London, 2016. arXiv preprint arXiv:1609.07480.

[72] T. Karvonen, G. Wynne, F. Tronarp, C.J. Oates, and S. Särkkä. Maximum likelihood estimation and uncertainty quantification for Gaussian process approximation of deterministic functions. *arXiv preprint arXiv:2001.10965*, 2020.

[73] J. Kim. *wrn-tensorflow*, 2018. https://github.com/dalgu90/wrn-tensorflow (accessed January, 2020).

[74] Y. Kim and K. Cho. Sea level rise around Korea: Analysis of tide gauge station data with the ensemble empirical mode decomposition method. *Journal of Hydro-environment Research*, 11:138–145, June 2016. https://doi.org/10.1016/j.jher.2014.12.002.

[75] G.S. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.*, 41(2):495–502, 04 1970. https://doi.org/10.1214/aoms/1177697089.

[76] S. Kizhner, K. Blank, T. Flatley, N.E. Huang, D. Patrick, and P. Hestnes. On the Hilbert-Huang transform theoretical developments. In *2006 IEEE Aerospace Conference*.

[77] R. Kohn, C.F. Ansley, and D. Tharm. The performance of cross-validation and maximum likelihood estimators of spline smoothing parameters. *Journal of the American Statistical Association*, 86(416):1042–1050, 1991. https://doi.org/10.1080/01621459.1991.10475150.

[78] D.Y. Kolotkov, S.A. Anfinogentov, and V.M. Nakariakov. Empirical mode decomposition analysis of random processes in the solar atmosphere. *Astronomy and Astrophysics*, 592(A153), 2016. https://doi.org/10.1051/0004-6361/201628306.

[79] D.G. Krige. *A statistical approach to some mine valuation and allied problems on the Witwatersrand*. PhD thesis, University of the Witwatersrand, 1951.

[80] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.222.9220.

[81] M. Kuntz and M. Helbich. Geostatistical mapping of real estate prices: an empirical comparison of kriging and cokriging. *International Journal of Geographical Information Science*, 28(9):1904–1921, 2014. https://doi.org/10.1080/13658816.2014.906041.

[82] M.W.Y. Lam. TLGProb: Two-layer Gaussian process regression model for winning probability calculation in two-team sports. In *International Conference on Artificial Intelligence and Soft Computing*, pages 280–291. Springer, 2017. https://doi.org/10.1007/978-3-319-59060-8_26.

[83] Y. LeCun, C. Cortes, and J.B. Christopher. The MNIST database of handwritten digits. 1998. http://yhann.lecun.com/exdb/mnist.

[84] T. Lee and T.B.M.J Ouarda. Prediction of climate nonstationary oscillation processes with empirical mode decomposition. *Journal of Geophysics Research*, 116, 2011. https://doi.org/10.1029/2010JD015142.

[85] H. Liu, Y. Ong, and J. Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems*, 2020. https://doi.org/10.1109/TNNLS.2019.2957109.

[86] P.J. Liu. Using Gaussian process regression to denoise images and remove artefacts from microarray data. Master's thesis, University of Toronto, 2007.

[87] S. Maji, A.C. Berg, and J. Malik. Efficient classification for additive kernel SVMs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):66–77, 2013. https://doi.org/10.1109/TPAMI.2012.62.

[88] O. Marinoni. Improving geological models using a combined ordinary–indicator kriging approach. *Engineering Geology*, 69(1-2):37–45, 2003. https://doi.org/10.1016/S0013-7952(02)00246-6.

[89] S. Meignen, T. Oberlin, and D.-H. Pham. Synchrosqueezing transforms: From low- to high-frequency modulations and perspectives. *Comptes Rendus Physique*, 20:446–460, 2019. https://doi.org/10.1016/j.crhy.2019.07.001.

[90] S. Meignen and V. Perrier. A new formulation for empirical mode decomposition based on constrained optimization. *IEEE Signal Processing Letters*, 14(12):932–935, 2007. http://doi.org/10.1109/LSP.2007.904706.

[91] A. Mert and A. Akan. Emotion recognition from EEG signals by using multivariate empirical mode decomposition. *Pattern Analysis and Applications*, 21(1):81–89, 2018. https://doi.org/10.1007/s10044-016-0567-6.

[92] C. A. Micchelli and T. J. Rivlin. A survey of optimal recovery. In *Optimal Estimation in Approximation Theory*, pages 1–54. Springer, 1977. https://doi.org/10.1007/978-1-4684-2388-4_1.

[93] P.A. Muñoz-Gutiérrez, E. Giraldo, M. Bueno-López, and M. Molinas. Localization of active brain sources from EEG signals using empirical mode decomposition: A comparative study. *Frontiers in Integrative Neuroscience*, 12, 2018. https://doi.org/10.3389/fnint.2018.00055.

[94] T. Oberlin, S. Meignen, and V. Perrier. The Fourier-based synchrosqueezing transform. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 315–319. IEEE, 2014. https://doi.org/10.1109/ICASSP.2014.6853609.

[95] E.J. O'Brien, A. Malekjafarian, and A. González. Application of empirical mode decomposition to drive-by bridge damage detection. *European Journal of Mechanics - A/Solids*, 61:151–163, Jan-Feb 2017. https://doi.org/10.1016/j.euromechsol.2016.09.009.

[96] A. O'Hagan. Some Bayesian numerical analysis. *Bayesian Statistics*, 4:345–363, 1992.

[97] R.A. Olea. *Geostatistics for Engineers and Earth Scientists*. Springer Science & Business Media, 2012.

[98] M. Osborne, S. Roberts, A. Rogers, and N. Jennings. Real-time information processing of environmental sensor network data using Bayesian Gaussian processes. *ACM Transactions on Sensor Networks (TOSN)*, 9(1):1–32, 2012. https://doi.org/10.1145/2379799.2379800.

[99] H. Owhadi. Multigrid with rough coefficients and multiresolution operator decomposition from hierarchical information games. *SIAM Review*, 59(1):99–149, 2017. https://doi.org/10.1137/15M1013894.

[100] H. Owhadi and C. Scovel. Universal scalable robust solvers from computational information games and fast eigenspace adapted multiresolution analysis. *arXiv preprint arXiv:1703.10761*, 2017.

[101] H. Owhadi and C. Scovel. *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization: From a Game Theoretic Approach to Numerical Approximation and Algorithm Design*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2019.

[102] H. Owhadi, C. Scovel, and G.R. Yoo. Kernel Mode Decomposition and programmable/interpretable regression networks. *arXiv preprint arXiv:1907.08592*, 2019.

[103] H. Owhadi and G.R. Yoo. Kernel Flows: From learning kernels from data into the abyss. *Journal of Computational Physics*, 389:22–47, 2019. https://doi.org/10.1016/j.jcp.2019.03.040.

[104] H. Owhadi and L. Zhang. Gamblets for opening the complexity-bottleneck of implicit schemes for hyperbolic and parabolic ODEs/PDEs with rough coefficients. *Journal of Computational Physics*, 347:99–128, 2017. https://doi.org/10.1016/j.jcp.2017.06.037.

[105] E. Pardo-Igúzquiza and P.A. Dowd. Empirical maximum likelihood kriging: The general case. *Mathematical Geology*, 37(5):477–492, 2005. https://doi.org/10.1007/s11004-005-6665-4.

[106] N. Pastorello, Forbes D.A., C. Foster, J.P. Brodie, C. Usher, A.J. Romanowsky, J. Strader, and J.A. Arnold. The SLUGGS survey: exploring the metallicity gradients of nearby early-type galaxies to large radii. *Monthly Notices of the Royal Astronomical Society*, 442(2):1003–1039, 2014. https://doi.org/10.1093/mnras/stu937.

[107] D.-H. Pham and S. Meignen. High-order synchrosqueezing transform for multicomponent signals analysis– with an application to gravitational-wave signal. *IEEE Transactions on Signal Processing*, 65(12):3168–3178, 2017. https://doi.org/10.1109/TSP.2017.2686355.

[108] N. Pustelnik, P. Borgnat, and P. Flandrin. A multicomponent proximal algorithm for empirical mode decomposition. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 1880–1884. IEEE, 2012. https://ieeexplore.ieee.org/abstract/document/6334130.

[109] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, 2006. www.GaussianProcess.org/gpml.

[110] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do CIFAR-10 classifiers generalize to CIFAR-10? *arXiv preprint arXiv:1806.00451*, 2018.

[111] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to imagenet? *arXiv preprint arXiv:1902.10811*, 2019.

[112] G. Rilling and P. Flandrin. Sampling effects on the empirical mode decomposition. *Advances in Adapative Data Analysis*, 1(1):43–59, 2009. https://doi.org/10.1142/S1793536909000023.

[113] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984), 2013. https://doi.org/10.1098/rsta.2011.0550.

[114] E. Rodner, A. Freytag, P. Bodesheim, and J. Denzler. Large-scale Gaussian process classification with flexible adaptive histogram kernels. In *European Conference on Computer Vision*, pages 85–98. Springer, 2012. https://doi.org/10.1007/978-3-642-33765-9_7.

[115] R.E. Rossi, J.L. Dungan, and L.R. Beck. Kriging in the shadows: Geostatistical interpolation for remote sensing. *Remote Sensing of Environment*, 49(1):32–40, 1994. https://doi.org/10.1016/0034-4257(94)90057-4.

[116] A. Rozza, G. Lombardi, C. Ceruti, E. Casiraghi, and P. Campadelli. Novel high intrinsic dimensionality estimators. *Machine learning*, 89(1-2):37–65, 2012. https://doi.org/10.1007/s10994-012-5294-7.

[117] S. Sahoo, M. Mohanty, S. Behera, and S.K. Sabut. ECG beat classification using empirical mode decomposition and mixture of features. *Journal of Medical Engineering and Technology*, 41(8):652–661, 2017. https://doi.org/10.1080/03091902.2017.1394386.

[118] P. Schneider, N. Castell, M. Vogt, F.R. Dauge, W. Lahoz, and A. Bartonova. Mapping urban air quality in near real-time using observations from low-cost sensors and model information. *Environment International*, 106:234–247, 2017. https://doi.org/10.1016/j.envint.2017.05.005.

[119] F. Schäfer, T.J Sullivan, and H. Owhadi. Compression, inversion, and approximate PCA of dense kernel matrices at near-linear computational complexity. *arXiv preprint arXiv:1706.02205*, 2017.

[120] M. Seeger. Relationships between Gaussian processes, support vector machines and smoothing splines. *Machine Learning*, 2000.

[121] K.D. Seger. An empirical mode decomposition-based detection and classification approach for marine mammal vocal signals. *The Journal of the Acoustical Society of America*, 144(6):3181–3190, 2018. https://doi.org/10.1121/1.5067389.

[122] P.Y. Simard, D. Steinkraus, and J. Platt. Best practices for Convolutional Neural Networks applied to visual document analysis. Institute of Electrical and Electronics Engineers, Inc., August 2003. https://www.microsoft.com/en-us/research/publication/best-practices-for-convolutional-neural-networks-applied-to-visual-document-analysis/.

[123] S. Spigler, M. Geiger, and M. Wyart. Asymptotic learning curves of kernel methods: empirical data vs teacher-student paradigm. *arXiv preprint arXiv:1905.10843*, 2019.

[124] D.A. Sprecher. On the structure of continuous functions of several variables. *Transactions of the American Mathematical Society*, 115:340–355, 1965.

[125] Srivastava, G. N., Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, Jun 2014. https://dl.acm.org/doi/abs/10.5555/2627435.2670313.

[126] M.L. Stein. Asymptotically efficient prediction of a random field with a misspecified covariance function. *The Annals of Statistics*, 16:55–63, 1988. https://www.jstor.org/stable/2241422.

[127] M.L. Stein. Bounds on the efficiency of linear predictions using an incorrect covariance function. *The Annals of Statistics*, 18:1116–1138, 1990. https://www.jstor.org/stable/2242045.

[128] M.L. Stein. A comparison of generalized cross validation and modified maximum likelihood for estimating the parameters of a stochastic process. *The Annals of Statistics*, 18:1139–1157, 1990. https://www.jstor.org/stable/2242046.

[129] M.L. Stein. Uniform asymptotic optimality of linear predictions of a random field using an incorrect second-order structure. *The Annals of Statistics*, 18:850–872, 1990. https://www.jstor.org/stable/2242137.

[130] I. Steinwart and A. Christmann. *Support Vector Machines*. Springer Science & Business Media, New York City, NY, 2008.

[131] C.J. Stone. Additive regression and other nonparamentric models. *The Annals of Statistics*, 13(2):689–705, 1985. www.jstor.org/stable/2241204.

[132] A.M. Stuart. Inverse problems: a Bayesian perspective. *Acta Numerica*, 19:451, 2010. https://doi.org/10.1017/S0962492910000061.

[133] A.L. Teckentrup. Convergence of Gaussian process regression with estimated hyper-parameters and applications in Bayesian inverse problems. *arXiv preprint arXiv:1909.00232*, 2019.

[134] G. Thakur and H.-T. Wu. Synchrosqueezing-based recovery of instantaneous frequency from nonuniform samples. *SIAM Journal on Mathematical Analysis*, 43(5):2078–2095, 2011. https://doi.org/10.1137/100798818.

[135] A. Torralba, R. Fergus, and W.T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008. https://doi.org/10.1109/TPAMI.2008.128.

[136] M. van der Wilk, C.E. Rasmussen, and J. Hensman. Convolutional Gaussian processes. In *Advances in Neural Information Processing Systems*, pages 2849–2858. 2017.

[137] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):480–492, 2012. https://doi.org/10.1109/TPAMI.2011.153.

[138] G. Wahba. A comparison of GCV and GML for choosing the smooth spline smoothing problem. *The Annals of Statistics*, 13(4):1378–1402, Dec. 1985. https://www.jstor.org/stable/2241360.

[139] C.K.I. Williams and C.E. Rasmussen. Gaussian processes for regression. In *Advances in neural information processing systems*, pages 514–520, 1996.

[140] A.G. Wilson and H. Nickisch. Kernel interpolation for scalable structured Gaussian processes (KISS-GP). In *International Conference on Machine Learning*, pages 1775–1784, 2015.

[141] L.-C. Wu, C.C. Kao, T.-W. Hsu, K.-C. Jao, and Y.-F. Wang. Ensemble empirical mode decomposition on storm surge separation from sea level data. *Coastal Engineering Journal*, 53(3):223–243, 2011. https://doi.org/10.1142/S0578563411002343.

[142] Z. Wu and N.E. Huang. Ensemble empirical mode decomposition: a noise-assisted data analysis method. *Advances in Adaptive Data Analysis*, 1:1–41, 2009. https://doi.org/10.1142/S1793536909000047.

[143] G. Wynne, F-X. Briol, and M. Girolami. Convergence guarantees for Gaussian process approximations under several observation models. *arXiv preprint arXiv:2001.10818*, 2020.

[144] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[145] H Xie, L. Zhang, and H. Owhadi. Fast eigenpairs computation with operator adapted wavelets and hierarchical subspace correction. *SIAM Journal on Numerical Analysis*, 57:2519–2550, 2019. https://doi.org/10.1137/18M1194079.

[146] Y.-L. Xu and J. Chen. Structural damage detection using empirical mode decomposition: experimental investigation. *Journal of Engineering Mechanics*, 130:1279–1288, Nov 2004. https://doi.org/10.1061/(ASCE)0733-9399(2004)130:11(1279).

[147] Y.-J. Xue, J.-X. Cao, G.L. Zhang, H.-K. Du, Z. Wen, X.-H. Zeng, and F. Zou. Application of local wave decomposition in seismic signal processing.

In T. Zouaghi, editor, *Earthquakes*, chapter 2. IntechOpen, Rijeka, 2017. https://doi.org/10.5772/65297.

[148] Y.-J. Xue, J. Zhang, Q. Chang, L.-P. Zhang, and F. Zou. Instantaneous frequency extraction using the emd-based wavelet ridge to reveal geological features. *Frontiers in Earth Science*, 6, 2018. https://doi.org/10.3389/feart.2018.00065.

[149] C. Yadav and L. Bottou. Cold case: The lost MNIST digits. In *Advances in Neural Information Processing Systems*, pages 13443–13452, 2019. http://papers.nips.cc/paper/9500-cold-case-the-lost-mnist-digits.pdf.

[150] A.C. Yang, J.-L. Fuh, N.E. Huang, B.-C. Shia, and S.-J. Wang. Patients with migraine are right about their perception of temperature as a trigger: time series analysis of headache diary data. *The Journal of Headache and Pain*, 16(1):1–7, 2015. https://doi.org/10.1186/s10194-015-0533-5.

[151] A.C. Yang, J.L. Fuh, N.E. Huang, B.C. Shia, C.K. Peng, and S.J. Wang. Temporal associations between weather and headache: analysis by empirical mode decomposition. *PLoS One*, 6(1), Jan 2011. https://doi.org/10.1371/journal.pone.0014612.

[152] D. Yinfeng, L. Yingmin, and L. Ming. Structural damage detection using empirical-mode decomposition and vector autoregressive moving average model. *Soil Dynamics and Earthquake Engineering*, 30:133–145, Mar 2010. https://doi.org/10.1016/j.soildyn.2009.10.002.

[153] G.R. Yoo and H. Owhadi. De-noising by thresholding operator adapted wavelets. *Statistics and Computing*, 29(6):1185–1201, 2019. https://doi.org/10.1007/s11222-019-09893-x.

[154] G.R. Yoo and H. Owhadi. Deep regularization and direct training of the inner layers of Neural Networks with Kernel Flows. *arXiv preprint arXiv:2002.08335*, 2020.

[155] S. Zagoruyko and N. Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016. https://dx.doi.org/10.5244/C.30.87.

[156] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. 2016.

[157] F. Zhu, T. Carpenter, D.R. Gonzalez, M. Atkinson, and J. Wardlaw. Computed tomography perfusion imaging denoising using Gaussian process regression. *Physics in Medicine & Biology*, 57(12):N183, 2012. https://doi.org/10.1088/0031-9155/57/12/N183.

[158] W. Zvarevashe, S. Kirshnannair, and V. Sivakumar. Analysis of rainfall and temperature data using ensemble empirical mode decomposition. *Data Science Journal*, 18(1), 2019. http://doi.org/10.5334/dsj-2019-046.