Foundations of deep convolutional models through kernel methods

Alberto Bietti

Communauté
UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTÉ UNIVERSITÉ GRENOBLE ALPES

Spécialité : Mathématiques appliquées

Arrêté ministériel : 25 mai 2016

Présentée par

Alberto BIETTI

Thèse dirigée par **Julien MAIRAL**, chercheur, Communauté Université Grenoble Alpes

préparée au sein du **Laboratoire Jean Kuntzmann** dans **l'École Doctorale Mathématiques, Sciences et technologies de l'information, Informatique**

Méthodes à noyaux pour les réseaux convolutionnels profonds

Foundations of deep convolutional models through kernel methods

Thèse soutenue publiquement le **27 novembre 2019**, devant le jury composé de :

Monsieur JULIEN MAIRAL
CHARGE DE RECHERCHE, INRIA CENTRE DE GRENOBLE RHÔNE-ALPES, Directeur de thèse
Monsieur STEPHANE MALLAT
PROFESSEUR, COLLEGE DE FRANCE - PARIS, Rapporteur
Monsieur LORENZO ROSASCO
PROFESSEUR ASSOCIE, UNIVERSITE DE GÊNES - ITALIE, Rapporteur
Madame FLORENCE D'ALCHE-BUC
PROFESSEUR, TELECOM PARISTECH, Président
Monsieur JEAN-PHILIPPE VERT
PROFESSEUR, MINES PARISTECH, Examinateur
Monsieur JOAN BRUNA
PROFESSEUR ASSISTANT, UNIVERSITE DE NEW YORK - ETATS-UNIS, Examinateur

Abstract

The increased availability of large amounts of data, from images in social networks, speech waveforms from mobile devices, and large text corpuses, to genomic and medical data, has led to a surge of machine learning techniques. Such methods exploit statistical patterns in these large datasets for making accurate predictions on new data. In recent years, deep learning systems have emerged as a remarkably successful class of machine learning algorithms, which rely on gradient-based methods for training multi-layer models that process data in a hierarchical manner. These methods have been particularly successful in tasks where the data consists of natural signals such as images or audio; this includes visual recognition, object detection or segmentation, and speech recognition.

For such tasks, deep learning methods often yield the best known empirical performance; yet, the high dimensionality of the data and large number of parameters of these models make them challenging to understand theoretically. Their success is often attributed in part to their ability to exploit useful structure in natural signals, such as local stationarity or invariance, for instance through choices of network architectures with convolution and pooling operations. However, such properties are still poorly understood from a theoretical standpoint, leading to a growing gap between the theory and practice of machine learning. This thesis is aimed towards bridging this gap, by studying spaces of functions which arise from given network architectures, with a focus on the convolutional case. Our study relies on kernel methods, by considering reproducing kernel Hilbert spaces (RKHSs) associated to certain kernels that are constructed hierarchically based on a given architecture. This allows us to precisely study smoothness, invariance, stability to deformations, and approximation properties of functions in the RKHS. These representation properties are also linked with optimization questions when training deep networks with gradient methods in some over-parameterized regimes where such kernels arise. They also suggest new practical regularization strategies for obtaining better generalization performance on small datasets, and state-of-the-art performance for adversarial robustness on image tasks.

We begin with an analysis of the RKHS for a class of multi-layer convolutional kernels. In particular, we study properties of a representation of signals on continuous domains given by the kernel mapping of such kernels, showing that for appropriate choices of convolutional architectures, they are near-invariant to translations or other groups of transformations, stable to the action of diffeomorphisms on the input signal, and preserve signal information when appropriately discretized. We further show that certain classes of generic convolutional networks with smooth activations are contained in the corresponding RKHS, and characterize their RKHS norm, which acts as a measure of complexity, and additionally controls smoothness and stability.

The next contribution introduces a new family of practical regularization methods

for deep (convolutional) networks, by viewing such models as functions in the RKHS of a certain multi-layer kernel, and approximating its RKHS norm. We show that the obtained regularizers are effective on various tasks where only small amounts of labeled data are available, on visual recognition and biological datasets. These strategies also lead to state-of-the-art performance for obtaining models that are robust to adversarial perturbations on the CIFAR10 image dataset. Additionally, our viewpoint provides new theoretical insights and guarantees, including generalization bounds for robust learning with adversarial perturbations.

We then study generalization properties of optimization algorithms when training deep neural networks in a certain over-parameterized regime. Indeed, learning with gradient descent in such a regime corresponds to a kernel method in the infinite-width limit, with a kernel determined at initialization which then controls generalization properties. We study such kernels and their RKHS for two-layer networks and deep convolutional models, and show different trade-offs between smoothness and approximation properties, compared to other kernels obtained when only training the weights in the last layer.

Finally, we consider optimization problems that arise from a different approach to stability and regularization in machine learning, namely the commonly used data augmentation strategy. In such settings, the objective function is a sum over the training set of expectations over perturbations. We derive a new stochastic optimization algorithm which always outperforms stochastic gradient descent in the strongly convex case, with a dependence on a much smaller gradient variance quantity, that only depends on the variance due to perturbations on a single example, rather than across the entire dataset.

Keywords: machine learning, deep learning, kernel methods, convolutional networks, stability, regularization, optimization.

Résumé

La disponibilité de quantités massives de données, comme des images dans les réseaux sociaux, des signaux audio de téléphones mobiles, ou des données génomiques ou médicales, a accéléré le développement des techniques d'apprentissage automatique. Ces méthodes exploitent des motifs statistiques dans ces grandes bases de données pour effectuer de bonnes prédictions sur des nouvelles images, signaux, ou séquences de protéines. Récemment, les systèmes d'apprentissage profond ont émergé comme des algorithmes d'apprentissage très efficaces. Ces modèles multi-couche effectuent leurs prédictions de façon hiérarchique, et peuvent être entraînés à très grande échelle avec des méthodes de gradient. Leur succès a été particulièrement marqué lorsque les données sont des signaux naturals comme des images ou des signaux audio, pour des tâches comme la reconnaissance visuelle, la détection d'objects, ou la reconnaissance de la parole.

Pour de telles tâches, l'apprentissage profond donne souvent la meilleure performance empirique, mais leur compréhension théorique reste difficile à cause du grand nombre de paramètres, et de la grande dimension des données. Leur succès est souvent attribué à leur capacité d'exploiter des structures des signaux naturels, par exemple en apprenant des représentations invariantes et multi-échelle de signaux naturels à travers un bon choix d'architecture, par exemple avec des convolutions et des opérations de pooling. Néanmoins, ces propriétés sont encore mal comprises théoriquement, et l'écart entre la théorique et pratique en apprentissage continue à augmenter. Cette thèse vise à réduire cet écart grâce à l'étude d'espaces de fonctions qui surviennent à partir d'une certaine architecture, en particulier pour les architectures convolutives. Notre approche se base sur les méthodes à noyaux, et considère des espaces de Hilbert à noyaux reproduisants (RKHS) associés à certains noyaux construits de façon hiérarchique selon une architecture donnée. Cela nous permet d'étudier précisement des propriétés de régularité, d'invariance, de stabilité aux déformations du signal, et d'approximation des fonctions du RKHS. Ces propriétés sur la représentation sont aussi liées à des questions d'optimisation pour l'entraînement de réseaux profonds à très grand nombre de neurones par descente de gradient, qui donnent lieu à de tels noyaux. Cette théorie suggère également des nouvelles stratégies pratiques de régularisation qui permettent d'obtenir une meilleure performance en généralisation pour des petits jeux de données, et une performance état-de-l'art pour la robustesse à des perturbations adversariales en vision.

Nous commençons par une analyse du RKHS pour une classe de noyaux convolutifs multi-couche. En particulier, nous étudions la représentation de signaux donnée par le "mapping" du noyau, et montrons que pour des bons choix d'architectures, celle-ci est quasi-invariante aux translations ou à d'autres groupes de transformations, stable à l'action de difféomorphismes sur le signal d'entrée, et préserve l'information du signal avec le bon sous-échantillonnage. Ensuite, nous montrons que le RKHS contient certaines

classes de réseaux convolutifs génériques avec des activations lisses, et caractérisons leur norme RKHS, qui joue le rôle d'une mesure de complexité, et contrôle aussi la régularité et la stabilité des prédictions.

Notre deuxième contribution introduit une famille de méthodes pratiques de régularisation pour les réseaux (convolutifs) profonds, en voyant de tels modèles comme des fonctions du RKHS d'un certain noyaux multi-couche, et en approximant leur norme RKHS. Nous montrons que ces techniques de régularisation sont efficaces sur plusieurs tâches avec peu de données étiquetées, en reconnaissance d'image et en bioinformatique. Nous obtenons aussi une performance état-de-l'art pour obtenir des modèles robustes à des perturbations adversariales sur le jeu de données CIFAR10. De plus, notre point de vue fournit de nouvelles intuitions et garanties théoriques, telles que des bornes de généralisation pour l'apprentissage robuste avec perturbations adversariales.

Ensuite, nous étudions les propriétés de généralisation de l'optimisation de réseaux profonds dans une certain régime sur-parametrisé. En effet, l'apprentissage par descente de gradient dans ce regime correspond à une méthode à noyau dans la limite de nombre de neurones infini, avec un noyau déterminé par l'initialisation, qui contrôle ainsi les propriétés de généralisation. Nous analysons ces noyaux et leur RKHS pour des réseaux à deux couches ainsi que pour des réseaux convolutifs profonds, et montrons des compromis différents entre régularité et approximation, comparé à d'autres noyaux obtenus en entraînant la dernière couche tout en fixant les couches intermédiaires.

Enfin, nous traitons des problèmes d'optimisation qui surviennent lorsque l'on utilise une approche différente mais fréquente pour la stabilité et la régularisation en apprentissage, c'est à dire l'augmentation de données. Dans ce cadre, la fonction objectif est une somme sur les points d'entraînement d'espérances sur les perturbations aléatoires. Nous proposons une nouvel algorithme d'optimisation stochastique avec un taux de convergence toujours meilleur que celui du gradient stochastique dans le cas fortement convexe, qui dépend d'une quantité de variance beaucoup plus petite, dépendant uniquement de la variance induite sur les gradients par les perturbations sur un seul exemple, plutôt que sur l'ensemble des données.

Contents

Acknowledgements

The accomplishment of this thesis would not have been possible without the help of numerous people. Julien, I feel very lucky to have had you as an advisor. Your knowledge, your insight on many different problems, and your taste in research have been instrumental throughout my PhD and I will always strive for similarly high research standards. I am forever grateful for your guidance and your time during these formative years.

I am thankful to all the jury members for willing to evaluate my work, and in particular to Stéphane Mallat and Lorenzo Rosasco for accepting to review this manuscript. The Thoth team at Inria has provided an ideal environment for the last three years and I thank Cordelia, Jakob, Karteek, Jocelyn and Nathalie making it so. In Grenoble, I've enjoyed working with various collaborators on research or software projects, in particular with Grégoire Mialon, Dexiong Chen, Ghislain Durif, and more recently Houssam Zenati, Eustache Diemert and Matthieu Martin. I've also had the pleasure to spend three months at Microsoft Research in New York, where I enjoyed "baking" contextual bandit algorithms under the great mentorship of John Langford and Alekh Agarwal, who helped shape my interest for statistical learning theory and interactive learning.

The list of colleagues and friends that have made the last few years a memorable experience is long, and I hope those I missed will forgive me. To members of the Thoth team: Nikita and Konstantin for being great office mates and for teaching me the most essential russian phrases, Thomas and Mathilde for all the raclette and guacamole, Valentin for your reliable presence at the jazz jams, Dexiong for the green tea and the dumplings, Alex for the mornings in the snow powder, Adria for the many funny anecdotes of deep learning practice, Vlad and Daan for the entertaining discussions, Grégoire, Bruno, Houssam and Alexandre for the short but frequent visits full of conversations, and the list goes on, Andrei, Minttu, Xavier, Ricardo, Lina, Hongzhou, Ghislain, Pavel, Pauline, Shreyas, Vicky, Gregory, Nicolas, Philippe, Marco, Henrique. To friends that have been of good company during conferences, workshops, and travel: Arthur, Lénaïc, Gaëtan, Kevin, Mehdi, Matthew, Mathias, Yossi, Nicolas, Vincent, Thomas, and the "Quora ML mafia" Tudor, Ofir, Denis, Jack, Jerry, Matt. To the interns, postdocs and other members of MSR NY for fun discussions, in particular Chris, Dylan, Hoang, Chicheng, Nan, Steven, Hal, Miro, Rob, Jacob, Marco, Yann. For the good time spent in Grenoble, thanks Nil, Federico, Matteo, Giovanni, Lina, Gianluca, Nestor, Luciano. To musical friends, partners, and teachers in Grenoble, Léa, Hélène, Moana, Olivier P, Lucie, Olivier T, Christian, Audrey, Jo, Samson, Stéphane, Lionel.

To my parents, for their constant support and encouragement and for instilling curiousity in me. To Elettra, for showing me the virtues of being a scholar. To Mosa, for her endless patience and support from both sides of the ocean, in music and in life.

To Nunna, who always believed in me, and will be missed.

Chapter 1

Introduction

The goal of supervised machine learning is to develop algorithms that can automatically learn models from a set of labeled examples, in order to make predictions on new (unlabeled) examples. Compared to more traditional methods in statistics, this focus on prediction has led to many empirical successes of machine learning on complex tasks where both the data and the models are very high-dimensional. Examples of such tasks include object recognition or scene segmentation in images, speech recognition in audio signals, and natural language understanding, all of which are often labeled with the term *artificial intelligence* (AI).

At the core of many recent successes on these problems are *deep learning* models, which have emerged after a long history of developments in neural networks, dating back to Rosenblatt's perceptron (Rosenblatt, 1958). The first major success in this new era of deep networks is likely the work of Krizhevsky et al. (2012), which employs a few recent "engineering" developments for training deep neural networks, along with large amounts of training data and efficient computation. Nevertheless, these developments have largely been empirical in nature, and the widespread use of deep networks in later years has kept widening the gap between theory and practice in machine learning, with models that are perceived as clever but complex engineering black boxes developed through practical insights while the theoretical understanding of why they work lags far behind. Multiple contributions in this thesis are aimed at reducing this gap.

The possibility to learn something meaningful from data crucially relies on various forms of *simplicity*. The *no free lunch* theorem states that no single learning algorithm can succeed on all possible problems, and it is thus important to enforce a form of simplicity in the algorithm (a form of Occam's razor), typically by restricting the class of models to be learned, which may reflect prior knowledge about the problem being tackled. This is also referred to as *inductive bias*. In the context of neural networks, one form of simplicity is in the choice of *architecture*, such as using convolutional neural networks (LeCun et al., 1989) when learning from image data. Another form of simplicity is *smoothness*, for instance in the context of non-parametric regression, when learning functions with small Lipschitz constant or small norm in a reproducing kernel Hilbert space (RKHS, see, *e.g.*, Schölkopf and Smola, 2001; Berlinet and Thomas-Agnan, 2004). A third example is *sparsity*, which may seek models that only rely on a few relevant variables out of many available ones (see, *e.g.*, Hastie et al., 2015).

In the context of neural networks, the architecture of the network has often been

recognized to be useful for making learning more efficient, for instance by using convolutional architectures on image data (LeCun et al., 1989, 2015). For example, the effectiveness of convolutional architectures is often attributed in part to their ability to provide some translation invariance, and exploit local stationarity in the image domain at multiple scales. Yet, a precise theoretical study of such properties for generic convolutional architectures was still missing before this thesis. One desirable property that goes beyond translation invariance is stability of predictions to small deformations of the input signal (*e.g.*, leading to a small change in handwriting on handwritten digits, which would preserve the label). A formal study of such a property was initiated in the context of the wavelet-based scattering transform (Mallat, 2012; Bruna and Mallat, 2013), which takes the form of a deep convolutional network based on convolutions with wavelet filter banks, thus involving no learning per se. Other works have constructed different architectures in order to extend invariance properties to different transformation groups, such as rotations, reflections, or scaling (Sifre and Mallat, 2013; Cohen and Welling, 2016).

Kernels methods have been extensively studied in machine learning, for learning non-linear functions defined on data with arbitrary structure, using techniques from linear models (see, *e.g.*, Schölkopf and Smola, 2001; Shawe-Taylor and Cristianini, 2004; Berlinet and Thomas-Agnan, 2004; Wahba, 1990). In this setting, the inductive bias of learning comes mainly from one mathematical object that determines similarities between two datapoints, namely a *positive definite kernel* function. This kernel defines a Hilbert space of functions (the RKHS) that one may learn, and one can study smoothness and approximation properties of these functions independently of the learning task. The norm of these functions then provides a useful notion of model complexity which can be used for model selection as well as for controlling smoothness. Hence, when dealing with structured data such as images, one may include desired prior knowledge in the choice of kernel; for instance, one can encourage invariance to various transformation groups, or small shifts, by constructing a kernel that appropriately pools over such transformations, as studied, *e.g.*, by Anselmi et al. (2016); Haasdonk and Burkhardt (2007); Mairal et al. (2014); Raj et al. (2017). It is also possible to introduce hierarchical structure in a kernel, by defining a kernel hierarchically (*e.g.* Cho and Saul, 2009; Mairal et al., 2014; Steinwart et al., 2016). These properties of kernel methods make them a useful framework for studying architectures in deep learning, by defining hierarchical kernels that mimick a given architecture, as those of Cho and Saul (2009); Daniely et al. (2016); Mairal et al. (2014); Mairal (2016). In addition to providing explicit kernels that carry desired properties of deep architectures, such kernels also arise naturally when considering the behavior of very wide networks with certain initializations, a study initiated by Neal (1996) for two-layer networks and recently extended to deeper architectures (Lee et al., 2018; Matthews et al., 2018), or when considering their evolution under gradient descent (Jacot et al., 2018).

The work described in this thesis follows these main directions, by considering theoretical and practical questions related to deep (convolutional) networks through the lens of kernel methods. In particular, it provides a theoretical analysis of invariance and stability to deformations for convolutional architectures, notions of complexity for obtaining generalization guarantees, regularization algorithms for controlling model complexity or robustness, and a detailed study of kernels that arise from the optimization of certain

over-parameterized networks. These contributions are further described in Section 1.1. The rest of this chapter gives an overview of key concepts that arise in the next chapters, such as statistical supervised learning, kernel methods, invariance and robustness.

1.1 Contributions of the thesis

This thesis brings various contributions with regard to the study of invariance, stability and regularization in machine learning, with a particular focus on deep models viewed through kernel methods. We review these contributions hereafter.

- Chapter 2 formalizes a general class of multi-layer convolutional kernels and studies invariance, stability to deformations, and signal preservation properties of the corresponding kernel mapping. Some of these properties naturally extend to the empirically effective approximations of such convolutional kernels introduced by Mairal (2016). This is followed by an analysis of the models in the corresponding functional space, which provides theoretical insight into the model complexity and generalization properties of certain neural networks with smooth activations. This contribution is based on the following publications:

  A. Bietti and J. Mairal. Invariance and stability of deep convolutional representations. In *Advances in Neural Information Processing Systems (NIPS)*, 2017a

  A. Bietti and J. Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *Journal of Machine Learning Research (JMLR)*, 20(25):1–49, 2019a

- Chapter 3 provides an empirical algorithmic framework for regularization and robustness of neural networks, based on the insights of Chapter 2 on kernel-based complexity measures and stability. In particular, we view deep networks as (approximate) elements of a corresponding RKHS, and provide practical regularization strategies based on approximating its RKHS norm, which we find to be large unless explicitly controlled. Our methods are shown to perform well empirically, displaying significant performance gains when learning on small datasets, as well as for learning models that are robust to adversarial perturbations, where we obtain state-of-the-art performance on CIFAR10 with ℓ^2 perturbations. This contribution is based on the following paper:

  A. Bietti, G. Mialon, D. Chen, and J. Mairal. A kernel perspective for regularizing deep neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019

- Chapter 4 shifts the focus to the inductive bias of optimization algorithms on learning neural networks. In particular, it has been shown that in a certain over-parameterization limit, gradient descent may converge to the minimum-norm solution in a particular RKHS with a kernel derived at initialization, called the *neural tangent kernel*. We study properties of this kernel for various architectures and compare it to the kernels studied in Chapter 2, showing that it may have better

approximation properties (*i.e.*, a "larger" RKHS), at the cost of weaker stability and smoothness guarantees. This contribution is based on the following paper:

> A. Bietti and J. Mairal. On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019b

- Chapter 5 considers invariance and stability from a regularization perspective, namely through commonly used data augmentation strategies, which augment the training set with random perturbations of each example. We study optimization algorithms for the objective that arises in such settings, and introduce a novel algorithm based on variance reduction, which achieves similar convergence rates to stochastic gradient descent in strongly convex problems, but with better constants, which can be much smaller, depending on the variance induced by perturbations on the gradients on a *single* example. This contribution is based on the following paper:

  > A. Bietti and J. Mairal. Stochastic optimization with variance reduction for infinite datasets with finite sum structure. In *Advances in Neural Information Processing Systems (NIPS)*, 2017b

- Another contribution of this thesis, which is not included in this manuscript due to little overlap in topics, is the following paper, which is currently under revision in the Journal of Machine Learning Research:

  > A. Bietti, A. Agarwal, and J. Langford. A contextual bandit bake-off. *arXiv preprint arXiv:1802.04064*, 2018

  The paper considers the problem of contextual bandit learning, where a learning algorithm selects actions based on observed feature vectors ("contexts"), and only observes a loss for the chosen action, highlighting the need for exploration. Such a setting commonly arises in online recommendation systems or medical treatment assignment. We provide an extensive empirical evaluation of existing contextual bandit algorithms on a large corpus of over 500 datasets from supervised learning, where the contextual bandit setting is simulated by only revealing the loss for the chosen label. We make the surprising finding that a simple greedy baseline which performs no exploration other than implicitly through the diversity in the data is quite competitive on many datasets. This motivates heuristic modifications of other algorithms which achieve more efficient exploration and thus better practical performance on various datasets, as well as a new algorithm based on insights from active learning algorithms, for which we provide a detailed regret analysis.

Finally, we note that this thesis contributed a few software libraries related to the contributions above, which are presented in Chapter A.

1.2 Supervised Machine Learning and Inductive Bias

The goal of supervised machine learning is to learn to predict the labels of new examples, given a dataset of labeled examples. For example, given a dataset of images along with

a label indicating the category of object that they represent, one would like to find a prediction function or program which is able to predict the category of new images which were not seen before. The hope is that the learning algorithm will be able to generalize to new examples from the same distribution, by trying to find patterns in the training examples. At the core of learning algorithms are forms of simplicity, known as *inductive biases*, which encode prior knowledge about a given problem in order to allow efficient learning. This notion is a core concept throughout this thesis, which studies such inductive biases for convolutional architectures, and introduces new regularization methods for controlling the simplicity of learned models.

1.2.1 Supervised learning setting

The supervised learning problem can be formalized as follows. The learning algorithm has access to i.i.d. samples $S_n = \{(x_i, y_i)\}_{i=1,\dots,n} \subset \mathcal{X} \times \mathcal{Y}$, consisting of examples x_i (*e.g.*, images, or biological sequences) and labels y_i (*e.g.*, the category of object in the image, or the phenotype associated to the biological sequence) from a data distribution $\mathcal{D}$ with measure ρ, and the goal is to find a *prediction function* $f : \mathcal{X} \to \mathcal{Y}$ (or *hypothesis*) with small *expected loss* or *risk*

$$L(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(f(x), y)],$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is a loss function which depends on the task of interest. The quantity $L(f)$ is also referred to as *risk* or *generalization error*. Supervised learning typically one of the following tasks.

- In *classification*, the labels are discrete, with $\mathcal{Y} = \{1, \dots, K\}$ in a multiclass problem with K classes (or $K = 2$ in binary classification), and the loss is often chosen to be the 0-1 loss:
$$\ell(\hat{y}, y) = \mathbb{1}\{\hat{y} \neq y\}. \tag{1.1}$$
- In *regression*, the outputs are scalars ($\mathcal{Y} = \mathbb{R}$), and one may use for instance the squared loss:
$$\ell(\hat{y}, y) = (\hat{y} - y)^2, \tag{1.2}$$
though other choices are possible, such as the loss $|\hat{y} - y|^q$ for other values of q.

The *Bayes optimal predictor* is the minimizer of $L(f)$:

$$f^* \in \arg\min_{f:\mathcal{X}\to\mathcal{Y}} L(f). \tag{1.3}$$

A learning algorithm A attempts to find a prediction function $\hat{f}_n$ from the training samples S_n, such that $L(\hat{f}_n)$ is small and, ideally, close to the *Bayes risk*, $L(f^*)$.

1.2.2 Generalization and inductive bias

Given the setting we introduced, it is reasonable to ask when it is possible to guarantee that a learning algorithm works, in the sense that for any $\epsilon > 0$, one can guarantee that $L(\hat{f}_n) - L(f^*) \leq \epsilon$ for a large enough number of samples n, with high probability

(over the sampling of S_n, and any randomness in the algorithm).[1] It turns out that this is not possible in general, for instance if $\mathcal{X}$ is infinite (*e.g.*, $\mathcal{X} = \mathbb{R}^p$) in a classification task. This is a form of the *no free lunch theorem*, which shows that one can always find a distribution on which a given algorithm fails (see, *e.g.* Shalev-Shwartz and Ben-David, 2014, Corollary 5.2).

This suggests that we need to somehow limit the difficulty of the problem in order to enable learning, either by constraining the algorithm, or by making assumptions on the data distribution. The most common way to constrain the algorithm is to restrict the search of possible predictors $\hat{f}$ to a particular subset of functions, of *hypothesis class*, $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$, for instance linear predictors, decision trees of a certain depth, or neural networks with a specific architecture or number of weights. Such constraints are typically enforced by the user based on prior knowledge about the problem, and they form the *inductive bias* of learning. A focus of this thesis is on studying inductive biases that are useful for natural signal data, such as those induced by convolutional architectures in deep learning.

Approaches to generalization. In some frameworks, we would like the learner to succeed regardless of the distribution, as is the case in (agnostic) PAC learning, where the goal is to guarantee that for any $\epsilon > 0$, with n large enough, we have that for *any* data distribution,

$$L(\hat{f}_n) - \min_{f \in \mathcal{F}} L(f) \leq \epsilon, \tag{1.4}$$

with high probability. The number of samples needed to achieve this, as a function of ϵ, is called the *sample complexity*. Here, no assumption is made about the Bayes predictor f^*, and we simply want to do well compared to the best possible predictor in the hypothesis class $\mathcal{F}$. This is the basis of the *distribution-free* approach to learning (see, *e.g.*, Devroye et al., 1996; Vapnik, 2000; Shalev-Shwartz and Ben-David, 2014).

An alternative approach is to make modeling assumptions about the data distribution, for instance assuming a model of the form $y_i = f^*(x_i) + \epsilon_i$, where ϵ_i are i.i.d., zero-mean noise variables, and some assumptions are made on f^*, such as Lipschitzness, or membership to a given reproducing kernel Hilbert space. When using the squared loss, is is easy to see that the Bayes predictor (also known as the *regression function* in this case) is equal to f^*, and one would like to establish upper bounds of the form

$$L(\hat{f}_n) - L(f^*) \leq \psi_n,$$

which hold in expectation or with high probability, where ψ_n is the rate of convergence. The best possible rate achievable under these assumptions (that is, first taking the worst possible f^* for each algorithm, and then considering the best resulting algorithm) is called the *minimax rate*. This approach is the one considered in *non-parametric statistics* (see, *e.g.*, Tsybakov, 2008; Wainwright, 2019).

The curse of dimensionality. It is a well-known fact that in high dimensions, there is "exponentially more room", as can be seen for instance when considering how the volume of a cube of side r grows exponentially with the dimension, as r^p in dimension p. This fact makes various tasks intractable in high dimensions, and also affects the tractability of

[1] This is related to the PAC learning formalism (Valiant, 1984; Shalev-Shwartz and Ben-David, 2014).

learning when the data lies in a high-dimensional space, particularly when considering hypothesis classes that are too large, or assumptions on the data generating process which are too weak. In the context of classification, an example is the simple nearest neighbor classifier—where the output label is chosen to be the one of the closest training example in Euclidian distance—, which requires a sample complexity exponential in the dimension (*e.g.*, Shalev-Shwartz and Ben-David, 2014, Chapter 19). Another example is that of learning with Lipschitz functions: while the no free lunch theorem discussed above states that it is impossible to learn with all functions $f : \mathcal{X} = \mathbb{R}^p \to \mathcal{Y}$, one can show that it is possible to learn with Lipschitz functions, that is with a hypothesis class $\mathcal{F}_L = \{f : \forall x, y \in \mathcal{X}, |f(x) - f(y)| \leq L\|x - y\|\}$, however the sample complexity will grow exponentially with the dimension, something which is unavoidable in this setting (see, *e.g.*, von Luxburg and Bousquet, 2004; Wainwright, 2019). Thus, when data is high-dimensional, it is important to impose further assumptions on the function class, or on the regression function in a non-parametric setting. For natural signals such as images or audio, which are typically very high-dimensional, it is natural to consider certain invariance and stability properties in order to mitigate this dependence on dimension.

1.2.3 Bias-variance trade-offs

Recall that we would like to learn a predictor with small risk, and hopefully as close as possible to the Bayes risk $L(f^*)$, but that we need restrictions on the hypothesis class or data distribution in order to be able to learn at all, as explained in Section 1.2.2. Intuitively, we want to consider a class of functions that is large enough to contain f^* (or a close approximation thereof), but also small enough so as to make learning easier. This leads to a trade-off which we discuss in this section.

Estimation-approximation trade-off. When considering a hypothesis class $\mathcal{F}$, we have the following decomposition on the excess risk:

$$L(\hat{f}_n) - L(f^*) \leq \underbrace{L(\hat{f}_n) - \min_{f \in \mathcal{F}} L(f)}_{\text{estimation error}} + \underbrace{\min_{f \in \mathcal{F}} L(f) - L(f^*)}_{\text{approximation error}}. \tag{1.5}$$

The first term is a *variance* term that deals with the ability to learn the hypothesis class $\mathcal{F}$ from a finite number of samples n, as seen in (1.4), and is known as *estimation error*. It increases as the hypothesis class becomes larger, since this makes learning harder. The second term is a *bias* term called *approximation error*, and decreases as $\mathcal{F}$ gets larger, reaching zero once $\mathcal{F}$ is large enough to contain f^*. As an example, one could consider fitting polynomials of a certain degree, or functions in an RKHS ball with a certain radius. Varying the degree or the norm radius then provides a way to balance this trade-off. Choosing a good value for such parameters is known as *model selection*, and is often carried in practice through *cross-validation* on a held-out set.

Bias-variance decomposition for least squares. A different decomposition is often used in the context of non-parametric estimation, which is easiest to see when using the squared loss. In this case, we have $f^*(x) = \mathbb{E}_{\mathcal{D}}[y|x]$, and one can show that $L(f) - L(f^*) = \mathbb{E}_{\mathcal{D}}[(f(x) - f^*(x))^2] = \|f - f^*\|_\rho^2$, with the notation $\|\cdot\|_\rho = \|\cdot\|_{L^2(d\rho)}$, where ρ

is the data distribution. For an estimator $\hat{f}_n$ obtained from the sample S_n, we denote $\bar{f}_n = \mathbb{E}[\hat{f}_n | x_{1:n}]$. We then have the following decomposition on the (expected) excess risk:

$$\begin{aligned}\mathbb{E}_{S_n}[L(\hat{f}_n) - L(f^*)] &= \mathbb{E}_{S_n}[\|\hat{f}_n - \bar{f}_n + \bar{f}_n - f^*\|_\rho^2] \\ &= \underbrace{\mathbb{E}_{S_n}[\|\hat{f}_n - \bar{f}_n\|_\rho^2]}_{\text{variance}} + \underbrace{\mathbb{E}_{x_{1:n}}[\|\bar{f}_n - f^*\|_\rho^2]}_{\text{bias}}, \end{aligned} \tag{1.6}$$

since the cross term cancels out when taking conditional expectations on $x_{1:n}$. In this decomposition, both terms are concerned with estimation since they depend on the data through the estimator $\hat{f}_n$ or its conditional expectation $\bar{f}_n$, however, only the first term depends on the noise in the output variables y_i and thus is referred to as *variance*, since it captures how much the estimator may be overly fitting on this noise. In contrast, the *bias* term considers how far the estimator is from f^* once the effects of noise are averaged out. A richer set of predictors should allow a small bias term: for instance, if there is no noise then one can essentially directly measure f^* and interpolate, so that a richer class of functions will reduce this bias. If there is noise, however, more complex predictors may fit the noise too well and lead to high variance predictions, though we note that recent work has found that a perfect fit to even noisy data may be acceptable in certain settings (see, *e.g.* Bartlett et al., 2019; Belkin et al., 2018a, 2019). For common estimators, one may then evaluate these two terms as a function of some complexity/smoothness/regularization parameter and balance them in order to obtain a final rate of convergence. More details can be found in the monographs by Györfi et al. (2006); Tsybakov (2008); Wainwright (2019).

1.2.4 Complexity and generalization bounds

Given a learning algorithm, it is useful to establish upper bounds on its generalization performance in order to understand their behavior and under what conditions they may work well. In this section, we briefly present generalization bounds that are relevant in this thesis in the context of kernel methods.

Empirical risk minimization. In some cases, it will be useful to think of the estimator $\hat{f}_n$ as the *empirical risk minimizer* (ERM) for a function class $\mathcal{F}$, given by

$$\hat{f}_n \in \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \ell(f(x_i), y_i). \tag{1.7}$$

This typically leads to an optimization problem which can be solved in closed form or using iterative optimization algorithms for some function classes and loss functions. When $\mathcal{F}$ takes the form $\mathcal{F} = \{f \in \mathcal{H} : \Omega(f) \leq B\}$ for some function space $\mathcal{H}$ and complexity measure $\Omega(f)$ (*e.g.*, a ball in an RKHS, or the space of linear predictors with bounded ℓ^1 norm), then it may be more convenient to use a penalized formulation

$$\hat{f}_n \in \arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^n \ell(f(x_i), y_i) + \lambda \Omega(f), \tag{1.8}$$

which is equivalent to (1.7) for well-chosen λ when the problem is convex, by Lagrange duality.

In the context of classification with the 0/1 loss, it is usually computationally hard to solve (1.7) or (1.8) directly since the objective is non-convex and discontinuous. Then, it is common to use a *surrogate loss* on a scalar-valued prediction function f instead, typically a function $\bar{\ell}(\cdot,\cdot)$ convex in its first argument that is an upper bound on the 0/1 loss of the sign classifier $\operatorname{sign}(f(\cdot))$ (assuming binary classification with labels in $\{-1,1\}$), that is,

$$\ell_{0/1}(\operatorname{sign}(f(x)),y) \le \ell(f(x),y).$$

Common examples are the *logistic* loss $\ell(f(x),y) = \log(1+\exp(-yf(x)))$, whose multiclass extension is often used in deep learning under the name *cross-entropy* loss, or the *hinge* loss $\ell(f(x),y) = \max(0,1-yf(x))$, which is the basis of support vector machines (Schölkopf and Smola, 2001; Shawe-Taylor and Cristianini, 2004). With these choices, a bound on the expected surrogate loss immediately yields a bound on the expected 0/1 loss of the corresponding sign classifier. We note that it is also possible to bound the *excess* risk for 0/1 loss in terms of the excess risk for some convex surrogates (see, *e.g.*, Boucheron et al., 2005; Bartlett et al., 2006; Rosasco et al., 2004).

Bounding estimation error using uniform convergence. In the decomposition (1.5), the estimation error deals with the learning algorithm and the use of a finite sample. One basic approach to control it is through *uniform convergence*, which control maximal deviations between empirical and expected risk, for all functions in $\mathcal{F}$. Denoting $\hat{L}(f) := \frac{1}{n}\sum_{i=1}^n \ell(f(x_i),y_i)$ the empirical risk on the sample S_n, and $f_{\mathcal{F}} \in \arg\min_{f\in\mathcal{F}} L(f)$, the empirical risk minimizer (1.7) satisfies

$$\begin{aligned} L(\hat{f}_n) - L(f_{\mathcal{F}}) &= L(\hat{f}_n) - \hat{L}(\hat{f}_n) + \hat{L}(\hat{f}_n) - \hat{L}(f_{\mathcal{F}}) + \hat{L}(f_{\mathcal{F}}) - L(f_{\mathcal{F}}) \\ &\le \hat{L}(\hat{f}_n) - \hat{L}(f_{\mathcal{F}}) + 2\sup_{f\in\mathcal{F}} |\hat{L}(f) - L(f)| \\ &\le 2\sup_{f\in\mathcal{F}} |L(f) - \hat{L}(f)|, \end{aligned}$$

where the last inequality follows from the definition of empirical risk minimizer. For many known function classes and losses, this supremum can be bounded with high probability over S_n. In particular, when the loss is uniformly bounded by a constant c, using concentration and a technique called symmetrization, one can show that the following holds with probability $1-\delta$ (*e.g.*, Boucheron et al., 2005; Shalev-Shwartz and Ben-David, 2014):

$$\sup_{f\in\mathcal{F}} |L(f) - \hat{L}(f)| \le 2\hat{R}_n(\ell\circ\mathcal{F}(S_n)) + c\sqrt{\frac{2\log\frac{2}{\delta}}{n}}, \tag{1.9}$$

where $\hat{R}_n(\ell\circ\mathcal{F}(S_n))$ is known as the *empirical Rademacher complexity* of the set $\ell\circ\mathcal{F}(S_n) = \{(\ell(f(x_1),y_1),\dots,\ell(f(x_n),y_n)) : f\in\mathcal{F}\}$ and is defined by

$$\hat{R}_n(A) = \mathbb{E}_\epsilon\left[\sup_{a\in A}\left|\frac{1}{n}\sum_{i=1}^n \epsilon_i a_i\right|\right],$$

where $\epsilon = (\epsilon_i)_i$ are i.i.d. Rademacher random variables, that is, uniform over $\{\pm 1\}$.

When ℓ is the 0/1 loss, we trivially have $c=1$ in (1.9), and the Rademacher complexity can be bounded using a combinatorial quantity known as the *VC dimension*—see,

e.g., Boucheron et al. (2005); Devroye et al. (1996); Shalev-Shwartz and Ben-David (2014); Vapnik (2000) for more details. Yet, this quantity typically grows with the number of parameters and is often unbounded for rich, non-parametric classes like kernel methods; additionally, the resulting bound we obtain on estimation error is for the empirical risk minimizer, which is typically intractable to obtain with the 0/1 loss.

In contrast, when $\ell(\cdot, y)$ is L-Lipschitz for any y, then we have the upper bound $\hat{R}_n(\ell \circ \mathcal{F}(S_n)) \leq L\hat{R}_n(\mathcal{F}(x_{1:n}))$ known as contraction lemma (see, *e.g.* Bartlett and Mendelson, 2002; Boucheron et al., 2005; Shalev-Shwartz and Ben-David, 2014), where

$$\hat{R}_n(\mathcal{F}(x_{1:n})) = \mathbb{E}_\epsilon \left[\sup_{f \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^{n} \epsilon_i f(x_i) \right| \right] \tag{1.10}$$

can now be bounded explicitly for certain classes such as kernel methods or linear models with bounds on certain norms. If we now consider an RKHS ball $\mathcal{F}_B = \{f \in \mathcal{H} : \|f\|_{\mathcal{H}} \leq B\}$ of a kernel K, and assume $K(x, x) \leq R^2$ for all $x \in \mathcal{X}$, one can show (Bartlett and Mendelson, 2002; Boucheron et al., 2005)

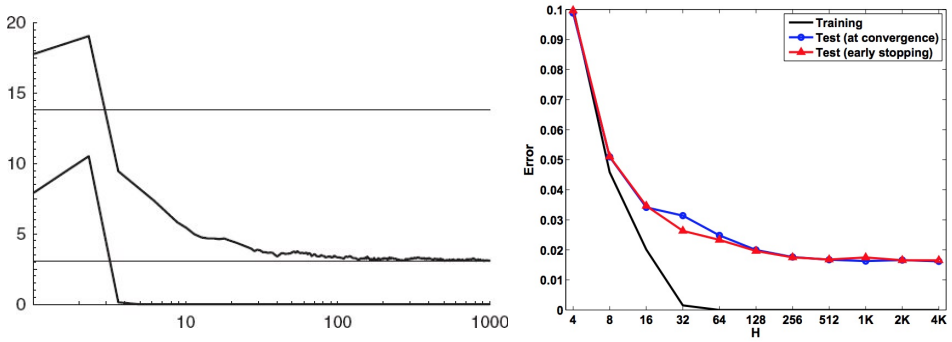
$$\hat{R}_n(\mathcal{F}(x_{1:n})) \leq \frac{BR}{\sqrt{n}}. \tag{1.11}$$

Similar bounds can be obtained for linear models with various ℓ_p norm constraints (Kakade et al., 2009). Assuming $\ell(0, y) \leq \ell_0$, we have the uniform bound $\ell(f(x_i), y) \leq \ell_0 + LBR$ on the loss, so that the bound (1.9), and thus the bound on the empirical risk minimizer is then of order

$$L(\hat{f}_n) - L(f_{\mathcal{F}}) \leq O\left((\ell_0 + LBR) \sqrt{\frac{\log(1/\delta)}{n}} \right).$$

Margin bounds for classification. A number of classification algorithms have been designed with the goal of maximizing the distance of points to the decision boundary—these are also known as *large margin* classifiers, and include the well-known support vector machine (SVM). Other than the hinge loss used in SVMs, other commonly used losses may implicitly encourage classifiers with large margins, such as the logistic loss or the exponential loss corresponding to some boosting classifiers (Schapire and Freund, 2012). Such margin-maximizing behavior can be used to explain the good generalization properties of such classifiers even when they perfectly fit the training data with models that are seemingly more complex, something which may be seen as surprising given that the algorithm is "overfitting" the training set (see Figure 1.1 for an illustration).

Indeed, while the size of these boosting or neural network models may be increasing, other notions of complexity, such as a norm-based quantity, may be well controlled. We now derive such a margin-based bound for the RKHS ball $\mathcal{F}_B$ considered above; see Boucheron et al. (2005); Koltchinskii and Panchenko (2002); Schapire et al. (1998); Anthony and Bartlett (2009); Kakade et al. (2009) for more details and examples for other function classes. Here, for a given margin $\gamma > 0$, we use a specific margin-sensitive surrogate loss instead of the loss used for obtaining $\hat{f}_n$, which helps characterize uniform convergence properties of the function class at this margin level. In particular, for binary classification with labels in $\{\pm 1\}$, we consider the "ramp" loss $\ell^\gamma(f(x), y) = \phi^\gamma(-yf(x))$



(a) boosting (Schapire et al., 1998)

(b) NN (Neyshabur et al., 2015b)

Figure 1.1: Training and test error curves for boosting or neural network classifiers with increasing complexity (number of base learners in boosting, or width of the neural network). In both cases, the test error keeps decreasing even after the training error is zero.

with

$$\phi^\gamma(u) = \begin{cases} 0, & \text{if } u \le -\gamma \\ 1, & \text{if } u \ge 0 \\ 1 + u/\gamma, & \text{otherwise,} \end{cases}$$

so that the loss is zero only when an example is classified with "confidence" at least γ. Given that this loss is upper bounded by 1 and is $(1/\gamma)$-Lipschitz, using (1.9) and the contraction lemma on Rademacher averages introduced above, the empirical and expected γ-losses satisfy, for all $f \in \mathcal{F}_B$,

$$L^\gamma(f) \le \hat{L}^\gamma(f) + \frac{2BR}{\gamma\sqrt{n}} + \sqrt{\frac{2\log\frac{2}{\delta}}{n}},$$

with probability $1-\delta$. Since $\ell_{0/1}(f(x),y) \le \ell^\gamma(f(x),y) \le \mathbb{1}\{yf(x) \le \gamma\}$, we have the following bound on the 0/1 loss of $\hat{f}_n$:

$$L_{0/1}(\hat{f}_n) \le \frac{1}{n}\sum_{i=1}^n \mathbb{1}\{y_i \hat{f}_n(x_i) \le \gamma\} + \frac{2BR}{\gamma\sqrt{n}} + \sqrt{\frac{2\log\frac{2}{\delta}}{n}}.$$

This is a form of *data-dependent* bound, as the bound depends on the *empirical margins* $(y_i\hat{f}_n(x_i))_i$. By taking a union bound over different values of γ and B, we may obtain the following bound which holds for all $\gamma > 0$ and for any classifier $\hat{f}_n$ obtained from the training data, without fixing a radius B in advance, at the cost of some additional logarithmic factors in δ, γ, and $\|\hat{f}_n\|_{\mathcal{H}}$ which are hidden here in the $\tilde{O}$ notation (see the proof of Proposition 3.1 in Chapter 3 for a detailed example): for all $\gamma > 0$,

$$L_{0/1}(\hat{f}_n) \le \frac{1}{n}\sum_{i=1}^n \mathbb{1}\{y_i \hat{f}_n(x_i) \le \gamma\} + \frac{4\|\hat{f}_n\|_{\mathcal{H}} R}{\gamma\sqrt{n}} + \tilde{O}\left(\frac{1}{\sqrt{n}}\right). \tag{1.12}$$

In particular, if the data is separated with some margin γ, then the first term vanishes and we obtain a bound of order $O(\|\hat{f}_n\|_{\mathcal{H}} R/\gamma\sqrt{n})$. A more refined bound may be obtained by trading off the first and second terms as γ varies, something which is often investigated empirically using the cumulative distribution of *normalized margins* $y_i \hat{f}_n(x_i)/\|\hat{f}_n\|_{\mathcal{H}}$; see Section 3.3.1 of Chapter 3 or Schapire et al. (1998); Bartlett et al. (2017) for examples.

Margin bounds similar in form to (1.12) have recently been used to study generalization for neural networks (trained in standard manners) based on various measures of complexity; see, *e.g.*, Bartlett et al. (2017); Neyshabur et al. (2018, 2019); Arora et al. (2018); Golowich et al. (2018). This thesis also contributes to this line of work with a complexity measure given by the norm in a particular RKHS, see Chapters 2 and 3.

More refined bounds and fast rates. One drawback of the bounds above is that they rely on properties that hold uniformly over the entire hypothesis class (due to uniform convergence bounds), and are thus unable to exploit favorable statistical properties which may hold only for functions that do well on the empirical sample, leading to rates of order $O(1/\sqrt{n})$ at best (these can be much worse for larger classes, *e.g.* of order $O(n^{-1/p})$ for Lipschitz functions in dimension p (von Luxburg and Bousquet, 2004)). In contrast, one can often obtain better rates (known as *fast rates*), for instance of order $O(1/n)$ in a linear least squares or ridge regression problem by making assumptions on the data distribution and by leveraging the exact form of the empirical risk minimizer (*e.g.*, Györfi et al., 2006; Hsu et al., 2014).

For more general settings such as generic non-parametric least-squares problems, or empirical risk minimization with other losses, one may obtain refined bounds by considering notions of complexity similar to the Rademacher complexities defined in (1.10), but that are evaluated on balls with varying radius around the Bayes predictor (in ℓ^2 distance on the sample), an approach known as *localization*. By analyzing the growth of such *localized* complexities in terms of this radius, one may obtain more precise bounds, including optimal rates for various non-parametric problems. For more details, see, *e.g.*, Bartlett et al. (2005); Koltchinskii (2006); Wainwright (2019). For the specific case of non-parametric least-squares in reproducing kernel Hilbert spaces, optimal convergence rates are often derived in terms of spectral properties of the kernel, which are known for many kernels used in practice (Cucker and Smale, 2002; Caponnetto and De Vito, 2007; Yao et al., 2007). For instance, the rates on estimation error may interpolate between $O(1/\sqrt{n})$ and $O(1/n)$ depending on how fast the eigenvalues of the kernel matrix decay. We discuss such spectral properties further in Section 1.3.

In the context of (binary) classification, one may achieve faster rates than $O(1/\sqrt{n})$ under various assumption. A simple example is when there is a perfect classifier for 0/1 loss, *i.e.*, $L(f^*) = 0$, in which case refined deviations bounds yield $O(1/n)$ convergence rates for the empirical risk minimizer (Boucheron et al., 2005; Vapnik and Chervonenkis, 1971). More generally, even if $L(f^*) > 0$, it has been noticed that fast rates are possible as long as the regression function $\eta(x) = P(y = 1|x)$ is well-behaved near the "critical threshold" 1/2, which defines the decision boundary for the Bayes-optimal classifier. In particular, if the data distribution puts little mass on points x such that $\eta(x)$ is near 1/2, then convergence can be much faster. This can be quantified with the so-called *margin conditions*, sometimes referred to as Tsybakov or Massart noise conditions (Mammen and Tsybakov, 1999; Massart and Nédélec, 2006). Such conditions help control localized

forms of complexity which can lead to fast rates for classification (Boucheron et al., 2005).

Interpolation. The empirical behavior of successful generalization despite zero training error, depicted in Figure 1.1, has challenged the theory of generalization for over twenty years. While the margin maximization view and the margin bounds similar to (1.12) provide a partial answer, recent observations have questioned their usefulness (Zhang et al., 2017a; Belkin et al., 2018b). In particular, deep learning practitioners typically train networks to near-zero training *loss*, even when the data is noisy. In such a situation, the learned prediction function interpolates the data and may rapidly grow in complexity with the number of examples, yet generalization still seems feasible. Recent papers have studied this question theoretically, showing that it is indeed possible to generalize (sometimes optimally) even in such interpolating regimes (Bartlett et al., 2019; Belkin et al., 2018a, 2019; Liang and Rakhlin, 2019).

1.2.5 Optimization and implicit regularization

The study of generalization is tied to a specific estimation procedure, and a standard choice is to consider the solution of a given optimization problem, such as empirical risk minimization, possibly with regularization through a penalty term or a constraint. Assuming such an optimization problem can be solved, the inductive bias of the procedure is then governed by the precise form of the minimizer, or by the geometry of the set of solutions. While such an analysis can be carried for many convex optimization problems thanks to optimality conditions, it is harder to establish this precisely for non-convex problems such as neural networks, since there may be many local minima and it is unclear a priori where gradient descent will end up. Even looser guarantees based on uniform convergence over a hypothesis class can be difficult to establish in this context, given that it is unclear what the precise hypothesis class is when training today's deep neural networks, which often have millions if not billions of parameters: simple combinatorial quantities depending on number of parameters, such as VC dimension, are likely too pessimistic, while other norm-based capacity measures can be defined in different ways (*e.g.*, Bartlett et al., 2017; Golowich et al., 2018; Neyshabur et al., 2015a, 2019), and it remains unclear if any of these is controlled in practice. These observations suggest that the optimization algorithm used for training a deep network plays a crucial role in determining the inductive bias of the learning problem.

Implicit bias of optimization. Iterative optimization algorithms have been known to provide a form of regularization, starting with early work in inverse problems (Landweber, 1951). In the context of learning, this regularization mechanism is known as *early stopping*, referring to the fact that we may benefit from stopping the optimization procedure early as a way to control the bias-variance trade-off. The generalization properties of such strategies have been studied, *e.g.*, for training kernel methods with the squared loss using gradient descent (Yao et al., 2007; Raskutti et al., 2014), or for boosting (Zhang et al., 2005) (which may be interpreted as a form of coordinate descent). For linear models $f(x) = \langle w, x \rangle$ with other losses such as the logistic or exponential loss, optimization procedures may not converge in the absence of a regularization term, for instance if the data is separable. In such cases, although the norm $\|w\|$ will grow unbounded, one can

show that the direction $\frac{w}{\|w\|}$ that gradient descent will converge to is the same as that of the maximum ℓ^2-margin solution (Soudry et al., 2018):

$$\begin{aligned} \min \quad & \|w\|_2 \\ \text{s.t.} \quad & y_i \langle w, x_i \rangle \geq 1, \quad i = 1, \dots, n. \end{aligned}$$

Thus, gradient descent implicitly biases the predictor towards the hard-margin SVM solution. The analysis is similar to the ℓ^1-margin-maximizing behavior of AdaBoost (Zhang et al., 2005; Telgarsky, 2013). Similar studies can be carried on simple neural network architectures without non-linearities (*i.e.*, *linear* networks) or matrix factorization models. Although the final model is effectively linear, the optimization problem is non-convex and under-determined, raising the question of which model will be found by the optimization procedure. In some cases, it can be shown that gradient descent then converges to a minimum norm solution, where the norm depends on the architecture/parameterization, such as the nuclear norm for matrix factorization, or a sparsity-inducing penalty in Fourier domain for linear convolutional networks (Gunasekar et al., 2017, 2018). Note that the geometry used for optimization (which is implicitly Euclidian when using gradient descent) may also affect the set of solutions found, and thus inductive biases, which may lead to different generalization properties (see, *e.g.*, Wilson et al., 2017).

Role of over-parameterization. Many successful deep network architectures tend to be widely over-parameterized, with many more parameters than the number of datapoints (Simonyan and Zisserman, 2014; Szegedy et al., 2016; Zagoruyko and Komodakis, 2016), which raises the question of its benefits from a theoretical perspective. One approach for studying over-parameterization is to consider limiting objects, that is, infinite-width networks, which can lead to clean theoretical models of complexity measures for neural networks, such as "convex neural networks" with sparsity-inducing norms (Bengio et al., 2006; Rosset et al., 2007; Bach, 2017a), or specific kernels (Neal, 1996; Cho and Saul, 2009; Lee et al., 2018; Matthews et al., 2018; Jacot et al., 2018).

For two-layer networks, such models can be defined through a measure on hidden units μ on $\mathbb{R}^p$:

$$f(x) = \int \sigma(\langle w, x \rangle) d\mu(w),$$

where σ is an activation function. Then, one can consider different norms on such functions through μ, such as L^1 or L^2 norms on $p(w)$ if p is a density of μ w.r.t. some base measure τ, *i.e.*, $d\mu(w) = p(w) d\tau(w)$. One can then study statistical aspects of such complexity measures (Bach, 2017a). In particular, the L^2 norm corresponds to a kernel method, while the L^1 norm (corresponding to a total variation norm on μ) provides benefits in terms of adapting to structure in the data thanks to its sparsity-promoting effect. However, it is still unclear if these are tied to common training strategies.

On the optimization side, over-parameterization has also been found beneficial in various regimes, by making the optimization landscape more benign (Soltanolkotabi et al., 2018), by enabling global convergence of gradient descent despite non-convexity (Chizat and Bach, 2018), or by introducing good local or global optima near initialization which can then be easily found (*e.g.*, Jacot et al., 2018; Li and Liang, 2018; Du et al., 2019b). When using the squared loss in the regime considered by this last scenario, then gradient descent attains the minimum-norm interpolating solution with respect to a kernel that

depends on the architecture at initialization, known as the *neural tangent kernel*. We study the inductive bias of such kernels in Chapter 4. A drawback of such an approach is that kernel methods may lead to poor sample complexity compared to more adaptive strategies such as the "convex neural network" approach described above, which may perform feature selection, similar to how deep networks are believed to discover useful features (*e.g.*, through Gabor-like filters at the early layers of CNNs). While such a sparsity-promoting regularization behavior is not currently well-understood for neural networks, it has been studied by Li et al. (2018) under specific assumptions for over-parameterized matrix sensing and two-layer networks with quadratic activations.

Learning as stochastic optimization. One may also view learning directly as an optimization problem, where the objective is the expected loss $L(f)$ and can be accessed through a *stochastic oracle* which may return for instance unbiased estimates of the gradient at a given query function, obtained for a given sample $(x, y) \sim \mathcal{D}$. For a linear model $f(x) = \langle w, x \rangle$ with a differentiable loss, we have $L(w) = \mathbb{E}_{x,y}[\ell(\langle w, x \rangle, y)]$, and such a stochastic first order oracle at a query point w would then return $g_t = \nabla_w \ell(\langle w, x_t \rangle, y_t)$ for $(x_t, y_t) \sim \mathcal{D}$, so that $\mathbb{E}_{(x_t,y_t)\sim\mathcal{D}}[g_t] = \nabla_w L(w)$. Then, an *optimization* guarantee $L(w_n) - \min_w L(w) \leq \epsilon_n$ after n queries to the oracle (corresponding to *one pass* over the training set) yields a *learning* guarantee on estimation error. The most common algorithm used in this setting is stochastic gradient descent (SGD), dating back to Robbins and Monro (1951), which performs the following update upon receiving a stochastic gradient estimate g_t at iteration t:

$$w_t = w_{t-1} - \eta_t g_t,$$

where η_t is a possibly decaying step-size, and the final estimate may involve an averaging of past iterates. Under various assumptions on the loss and the data, such as (strong) convexity, bounded gradients and bounded data, one may obtain convergence rates that are similar to the uniform convergence bounds on estimation error discussed in Section 1.2.4 (see, *e.g.*, Nemirovski et al., 2009; Bach and Moulines, 2011; Bubeck, 2015; Bottou et al., 2018). Under specific losses such as the squared loss, and more precise assumptions on the data distribution, one may also obtain faster rates through bias-variance decompositions (*e.g.*, Bach and Moulines, 2013; Dieuleveut et al., 2017; Neu and Rosasco, 2018). We note that different inductive biases may be used by changing the geometry of the gradient updates, typically by using a *mirror descent* or *dual averaging* framework (Nemirovski et al., 2009; Xiao, 2010; Bubeck, 2015).

Such stochastic gradient methods are attractive computationally for linear models since they only require to run one cheap iteration on each point of the dataset, which can often be sufficient for good learning performance, given the nature of learning as optimizing an expectation (Bottou and Bousquet, 2008). Nevertheless, optimizing a (regularized) empirical risk objective may still be beneficial in some problems (*e.g.*, when margin conditions may yield better statistical properties for the empirical risk minimizer), and there are efficient stochastic algorithms—based on a strategy known as *variance reduction*—for solving the resulting finite-sum optimization problems; see Chapter 5 and the references therein. In the context of deep learning models, SGD and its variants are the main optimization tool, yet they are used for many passes (*epochs*), and usually with large batches of datapoints, suggesting that the algorithm is minimizing

empirical risk as opposed to expected loss (this can also be seen through the fact that the loss can remain large on held-out data, as shown in Soudry et al., 2018, Figure 2).

1.3 Kernel Methods

In this section, we provide background on reproducing kernel Hilbert spaces (RKHSs) and corresponding positive-definite kernels, algorithms used with such kernels, and their generalization properties. More details no the topic can be found in the monographs (Schölkopf and Smola, 2001; Shawe-Taylor and Cristianini, 2004; Steinwart and Christmann, 2008; Berlinet and Thomas-Agnan, 2004; Wahba, 1990) or the lecture notes by Vert and Mairal (2017).

1.3.1 Kernels and reproducing kernel Hilbert spaces

Kernel methods in machine learning provide ways to learn non-linear models from rich functional spaces, in a tractable way through a kernel function. A *positive definite kernel* (p.d. kernel) is a symmetric function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that for any collection of points $x_1, \dots, x_n \in \mathcal{X}$ and scalars $a_1, \dots, a_n \in \mathbb{R}$, we have

$$\sum_{i,j} a_i a_j K(x_i, x_j) \geq 0.$$

Intuitively, $K(x, x')$ defines a *similarity measure* between two objects x and x' from a set $\mathcal{X}$ which may have an arbitrary structure. This definition is reminiscent of positive-definite matrices and inner products, and it is in fact possible to show that the such kernels correspond to an inner product on some (possibly non-linear) features:

Theorem 1.1 (Aronszajn (1950))**.** *A kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is positive definite if and only if there exists a Hilbert space H and $\phi : \mathcal{X} \to H$ such that*

$$K(x, x') = \langle \phi(x), \phi(x') \rangle_H, \tag{1.13}$$

for all $x, x' \in H$.

Such a feature map ϕ may define a space $\mathcal{H}$ in high dimensions (see Figure 1.2) in which a linear model may be effective, and this is what kernel methods do implicitly through evaluating the kernel functions on pairs of datapoints, as we discuss below. In particular, any explicitly defined feature map $\phi : \mathcal{X} \to \mathbb{R}^p$ defines a kernel (*e.g.*, one defined with various monomials in the coordinates of an input vector), but the power of kernels is that the feature map for a given kernel function may implicitly be infinite-dimensional, for instance for the Gaussian kernel $K(x, x') = e^{-\|x - x'\|^2}$.

Reproducing kernel Hilbert spaces. As we mentioned before, kernel methods enable us to learn models from certain functional spaces. These spaces are known as *reproducing kernel Hilbert spaces* (RKHSs) and are defined as follows.

Definition 1.2 (RKHS)**.** *A Hilbert space $\mathcal{H}$ is a RKHS if there is a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that:*

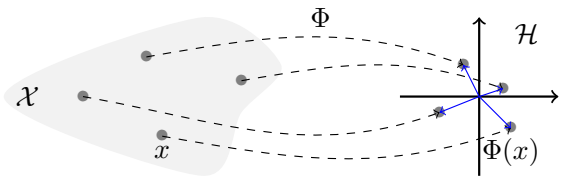


Figure 1.2: Illustration of the role of a kernel feature map Φ.

- *For all* $x \in \mathcal{X}$*, we have* $\Phi(x) := K(x, \cdot) \in \mathcal{H}$
- *For any* $f \in \mathcal{H}$ *and* $x \in \mathcal{X}$*, we have the* reproducing property*:*

$$f(x) = \langle f, \Phi(x) \rangle_{\mathcal{H}}. \tag{1.14}$$

If such a K *exists, it is known as the* reproducing kernel *of* $\mathcal{H}$*.*

As our phrasing suggests, it is possible to show that the reproducing kernel is unique for a given RKHS, and it is easy to see that it is p.d. since Φ defines a feature map (taking $f = \Phi(x')$ in (1.14)), often called the *canonical* feature map or kernel mapping. Conversely, one can show that a p.d. kernel defines a unique RKHS. This space characterizes the functions that are learned by kernel methods when using a p.d. kernel K, and hence controls the inductive bias, thus it is important to understand its properties.

Geometry and smoothness. A given choice of kernel defines a geometry for points in $\mathcal{X}$ through the Hilbert norm on its feature map Φ. The reproducing property and Cauchy-Schwarz then imply that the variations of a function f in $\mathcal{H}$ is controlled according to this geometry:

$$\begin{aligned} |f(x) - f(x')| &= |\langle f, \Phi(x) - \Phi(x') \rangle_{\mathcal{H}}| \\ &\leq \|f\|_{\mathcal{H}} \cdot \|\Phi(x) - \Phi(x')\|_{\mathcal{H}}. \end{aligned} \tag{1.15}$$

In particular, this shows that the RKHS norm of f controls the smoothness of its predictions, and can provide a useful mechanism to control complexity.

Note that the geometry of the kernel does not depend on the particular choice of feature map—indeed, if $\phi : \mathcal{X} \to H$ is another feature map for the kernel, we have $\|\Phi(x) - \Phi(x')\|_{\mathcal{H}}^2 = \|\phi(x) - \phi(x')\|_H^2$ by expanding the square. However, the norm of f is crucially defined in the RKHS. Indeed, one may define functions through a feature map ϕ to a different Hilbert space H from the RKHS $\mathcal{H}$, of the form $f(x) = \langle w, \phi(x) \rangle_H$ for some $w \in H$, but the norm $\|w\|_H$ in H may be larger than $\|f\|_{\mathcal{H}}$ even though both feature maps define the same kernel. For instance, if $\phi(x) := (x, x)^\top$, then the function $f(x) = \langle w, \phi(x) \rangle$ with $w = (a, -a)^\top$ is just the zero function, so that $\|f\|_{\mathcal{H}} = 0$, while we have $\|w\| = \sqrt{2}\|a\|$. This is formalized in the following theorem (see, *e.g.*, Saitoh, 1997, §2.1):

Theorem 1.3. *Let* $\phi : \mathcal{X} \to H$ *be a feature map to a Hilbert space* H*, and let* $K(x, x') := \langle \phi(x), \phi(x') \rangle_H$ *for* $x, x' \in \mathcal{X}$*. Let* $\mathcal{H}$ *be the linear subspace defined by*

$$\mathcal{H} := \{ f_w \ ; \ w \in H \} \quad s.t. \quad f_w : x \mapsto \langle w, \phi(x) \rangle_H,$$

and consider the norm

$$\|f_w\|_{\mathcal{H}}^2 := \inf_{w' \in H} \{\|w'\|_H^2 \quad s.t. \quad f_w = f_{w'}\}.$$

Then $\mathcal{H}$ is the reproducing kernel Hilbert space associated to kernel K.

1.3.2 Characterization of the RKHS

We saw that the geometry of the kernel mapping provides a control on the global smoothness of functions in the RKHS through (1.15). While this provides some insight into all RKHS functions of a given norm, a more precise description of the RKHS is desirable. For instance, one can show for various kernels that the kernel mapping is 1-Lipschitz (see, *e.g.*, Lemma 2.1 in Chapter 2), so that functions in the unit ball of their RKHS are 1-Lipschitz; yet, as discussed in Section 1.2.4, an RKHS ball is much smaller (in terms of complexity) than the set of all 1-Lipschitz functions, prompting us to characterize these functions more precisely.

When considering functions defined on a compact metric space $\mathcal{X}$, one approach is to study functions in the RKHS through their decomposition in an orthonormal basis of $L^2(\mathcal{X}, d\nu)$, for some non-negative measure ν on $\mathcal{X}$.[2] Indeed, for common choices of $\mathcal{X}$ and ν, such orthonormal bases are well studied and will lead to a good understanding of the RKHS. This can be achieved through a spectral decomposition of a linear operator associated to the kernel known as *integral operator* (*e.g.*, Cucker and Smale, 2002), in a similar fashion to spectral decompositions of positive semi-definite matrices. The integral operator T_K of a kernel K on $\mathcal{X}$ is defined for $f \in L^2(\mathcal{X}, d\nu)$ by

$$T_K f(x) := \int_{\mathcal{X}} K(x, y) f(y) d\nu(y). \tag{1.16}$$

Mercer's theorem then provides such a decomposition:

Theorem 1.4 (Mercer)**.** *If $\mathcal{X}$ is compact and K is continuous, then there exists an orthonormal basis $(\phi_i)_{i \in \mathbb{N}}$ of $L^2(\mathcal{X}, d\nu)$, and non-negative eigenvalues $(\mu_i)_{i \in \mathbb{N}}$ such that*

$$T_K \phi_i = \mu_i \phi_i, \quad i = 1, 2, \ldots.$$

Moreover, we have the expansion

$$K(x, y) = \sum_{i=1}^{\infty} \mu_i \phi_i(x) \phi_i(y),$$

where the convergence is absolute and uniform over $\mathcal{X} \times \mathcal{X}$. Then, one can verify that the RKHS $\mathcal{H}$ is given by

$$\mathcal{H} = \left\{ f = \sum_{i : \mu_i \neq 0} a_i \phi_i : a_i \in \mathbb{R}, \sum_{i : \mu_i \neq 0} \frac{a_i^2}{\mu_i} < \infty \right\},$$

where the norm of $f = \sum_{i:\mu_i \neq 0} a_i \phi_i$ is given by $\|f\|_{\mathcal{H}}^2 = \sum_{i:\mu_i \neq 0} \frac{a_i^2}{\mu_i}$.

[2] Recall that $L^2(\mathcal{X}, d\nu) = \{f : \mathcal{X} \to \mathbb{R} \text{ s.t. } \int_{\mathcal{X}} f^2 d\nu < \infty\}$.

One can in particular verify that Im $T_K = \mathcal{H}$ and that for $f \in \mathcal{H}$, we have

$$\|f\|_{\mathcal{H}} = \|T_K^{-1/2} f\|_{L^2(\mathcal{X}, d\nu)}, \tag{1.17}$$

where $T_K^{-1/2}$ is the negative square root of T_K (which is self-adjoint), which can be defined on Im $T_K = \mathcal{H}$. When using the RKHS norm as a regularizer, Eq. (1.17) provides an interpretation of $T_K^{-1/2}$ as a *regularization operator* in $L^2(\mathcal{X}, d\nu)$ (*e.g.*, Schölkopf and Smola, 2001). Here, this operator is diagonal in the basis $(\phi_i)_i$, multiplying by $1/\sqrt{\mu_i}$ the Fourier coefficient for ϕ_i, so that it will tend to attenuate most the components of f where μ_i is small. Conversely, if one is given a regularization operator D on $L^2(\mathcal{X}, d\nu)$, such as a differential operator, then one may find the RKHS $\mathcal{H}$ and corresponding reproducing kernel such that $\|f\|_{\mathcal{H}} = \|Df\|_{L^2(\mathcal{X}, d\nu)}$ by leveraging the reproducing property: $f(x) = \langle Df, DK(x, \cdot)\rangle_{L^2}$. Solving for $K(x, \cdot)$ corresponds to finding a *Green's function* for the operator D^*D (where D^* is the adjoint of D); see, *e.g.*, Schölkopf and Smola (2001, Chapter 4).

Example 1.5. *We provide a few canonical examples.*

- *(first-order Sobolev space) For the kernel $K(x, y) = \min(x, y)$ on $\mathcal{X} = [0, 1]$, the corresponding RKHS is a first-order Sobolev space which can also be expressed using a differential operator:*

$$\mathcal{H} = \{f : [0, 1] \to \mathbb{R}, f(0) = 0, f \text{ abs. cont. and } \int_0^1 f'(x)^2 dx < \infty\}.$$

  The Mercer decomposition when ν is the Lebesgue measure is given by

$$\phi_j(x) = \sin \frac{(2j-1)\pi x}{2}, \qquad \mu_j = \left(\frac{2}{(2j-1)\pi}\right)^2, \quad j = 1, 2, \ldots$$

- *(dot-product kernels on the sphere) For a kernel of the form $K(x, y) = \kappa(x^\top y)$ defined on the p-dimensional sphere $\mathcal{X} = \mathbb{S}^{p-1}$, when ν is the uniform measure on the sphere, then the Mercer decomposition can be given in the basis of* spherical harmonics. *A more detailed description for specific dot-product kernels is given in Chapter 4.*

- *(translation-invariant kernels) For kernels of the form $K(x, y) = \kappa(x - y)$, such as the Gaussian kernel $K(x, y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}$, the input space is often taken to be $\mathbb{R}^p$, in which case Mercer's theorem does not hold due to lack of compactness. However, a continuous decomposition can be given in terms of the Fourier transform $\hat{\kappa}$ of κ (a consequence of Bochner's theorem); this yields the RKHS norm*

$$\|f\|_{\mathcal{H}}^2 = \frac{1}{(2\pi)^p} \int_{\mathbb{R}^p} \frac{|\hat{f}(\omega)|^2}{\hat{\kappa}(\omega)} d\omega.$$

  For a Gaussian kernel, $\hat{\kappa}(\omega)$ decreases exponentially with $\|\omega\|^2$ as well as σ^2. When $\mathcal{X}$ is a compact subset of $\mathbb{R}^d$, Mercer's theorem applies, but the spectral decomposition is less straightforward, though it can be shown that the eigenvalues μ_i decay exponentially (Wainwright, 2019, Example 12.25), leading to a much smaller RKHS compared to the Sobolev space above.

The examples above show that many known kernels lead to spectral decompositions in known bases, such as Fourier or spherical harmonics. Then, one can relate the decay of eigenvalues μ_i to the regularity of functions in the RKHS; indeed it is well known that regularity is tightly linked to a certain decay of Fourier coefficients. In particular, the functions in the RKHS of the Gaussian kernel must be infinitely differentiable due to the exponential decay. One can also use these properties to provide approximation rates to certain classes of functions in terms of the radius of the RKHS ball (*e.g.*, Smale and Zhou, 2003); see also Bach (2017a) and Chapter 4 for such approximations for specific dot-product kernels arising from neural networks.

1.3.3 Algorithms

Kernel methods have been very successful in machine learning, because they lead to computationally tractable algorithms even though they are dealing with possibly infinite-dimensional function spaces. One way to see this is that any algorithm which can be written in terms of inner products between some (explicit) feature vectors $\phi(x_i)$ can be modified to use kernel evaluations instead, using (1.13), so that it will implicitly work with possibly infinite-dimensional features in the corresponding RKHS. This is known as the *kernel trick*, and applies particularly naturally when dealing with dual formulations of some optimization problems such as support vector machines, though it can also be readily applied in other contexts including clustering (leading to the Kernel K-means algorithm). We now present a more formal approach to learn with kernels.

The representer theorem. Fitting supervised machine learning models using kernel methods typically involves solving a regularized empirical risk minimization problem where the hypothesis space is a RKHS $\mathcal{H}$ associated to a kernel K:

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) + \lambda \|f\|_{\mathcal{H}}^2. \tag{1.18}$$

While solving this seems intractable at first sight since $\mathcal{H}$ may be infinite-dimensional, the *representer theorem*, due to Kimeldorf and Wahba (1971), states that any solution to (1.18) takes the form

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

for some $\alpha_1, \dots, \alpha_n \in \mathbb{R}$. Indeed, any non-zero component in f that is orthogonal to $span(\Phi(x_1), \dots, \Phi(x_n))$ does not change $f(x_i)$ in the first term of (1.18), but leads to a suboptimal regularization term, and hence must be zero at a minimizer. Denoting by $(K)_{ij} = K(x_i, x_j)$ the kernel matrix and letting $\alpha = (\alpha_1, \dots, \alpha_n)$, the problem then simplifies to

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

which can be solved using standard convex optimization tools when the loss is convex (*e.g.*, Schölkopf and Smola, 2001). For the squared loss $\ell(\hat{y}, y) = (\hat{y} - y)^2$, we have the explicit *kernel ridge regression* solution $\alpha = (K + n\lambda I_n)^{-1} y$, with $y = (y_1, \dots, y_n)^\top$.

This type of approach may also be applied to obtain non-linear formulations for other tasks, such as principal component analysis (Schölkopf et al., 1998) or independent component analysis (Bach and Jordan, 2002).

1.3.4 Statistical aspects

When learning with kernels, the choice of kernel affects the inductive bias of the learning procedure, and thus the resulting learning guarantees. If, for instance, one expects a problem to be solved well using a certain class of functions (*e.g.*, with certain smoothness properties), then using an RKHS adapted to these properties should lead to small estimation error. Kernel methods are particularly well-suited for such statistical analysis since the Hilbert structure of RKHSs makes this analysis similar to that of linear models, and particular ridge regression.

Rademacher complexity. For instance, the empirical Rademacher complexity of an RKHS ball $\mathcal{F}_B = \{f \in \mathcal{H}, \|f\|_{\mathcal{H}} \leq B\}$ on a sample $S_n = \{x_1, \dots, x_n\}$ with kernel matrix K can be upper bounded as follows (see also Section 1.2.4):

$$\begin{aligned}
\hat{R}_n(\mathcal{F}_B) &= \mathbb{E}_\epsilon \left[\sup_{f \in \mathcal{F}_B} \frac{1}{n} \sum_i \epsilon_i f(x_i) \right] \\
&= \frac{1}{n} \mathbb{E}_\epsilon \left[\sup_{f \in \mathcal{F}_B} \langle f, \sum_i \epsilon_i \Phi(x_i) \rangle_{\mathcal{H}} \right] \\
&\leq \frac{B}{n} \mathbb{E}_\epsilon \left[\sqrt{\epsilon^\top K \epsilon} \right] && \text{(Cauchy-Schwarz)} \\
&\leq \frac{B}{n} \sqrt{\mathbb{E}_\epsilon [\epsilon^\top K \epsilon]} && \text{(Jensen)} \\
&= \frac{B}{n} \sqrt{\operatorname{Tr}(K \, \mathbb{E}_\epsilon [\epsilon \epsilon^\top])} \\
&= \frac{B}{\sqrt{n}} \sqrt{\frac{1}{n} \sum_i K(x_i, x_i)},
\end{aligned}$$

where ϵ is a vector of i.i.d. Rademacher variables. This suggests in particular that regularization using the RKHS norm is useful for controlling estimation error. This result can be leveraged to obtain margin bounds on classification problems for any estimator learned from training data, as detailed in Section 1.2.4. We use this viewpoint to study regularization and generalization for neural networks through an appropriate RKHS in Chapters 2 and 3.

For settings where the estimator is the (regularized) empirical risk minimizer, one may obtain faster rates than the $O(1/\sqrt{n})$ rate above by using localized complexities, which will additionally display a dependence on the eigenvalues of the kernel matrix (see Bartlett et al., 2005; Wainwright, 2019). In particular, the rates depend on the rate of decay of these eigenvalues, and leads to a fast $O(1/n)$ rate for the Gaussian kernel where the decay is exponential, at least for a data distribution close to uniform. For the specific case of ridge regression, one can obtain such rates with a different approach which we now describe.

Kernel ridge regression and degrees of freedom. A more direct approach to analyzing the kernel ridge regression estimator is to consider a bias-variance decomposition of the form (1.6). For simplicity, here we describe such a decomposition in the *fixed design* setting, where the inputs $x_{1:n}$ are considered fixed, and we look at expected "in-sample" prediction error due to the noise in the outputs only. Consider a simple model where $y_i = f^*(x_i) + \epsilon_i$, where ϵ_i are i.i.d. $\mathcal{N}(0, \sigma^2 I)$ noise variables. Denoting $y_i^* = \mathbb{E}_\epsilon\, y_i = f^*(x_i)$, K the kernel matrix, and $\hat{y} = K(K + n\lambda I)^{-1} y$ the ridge regression predictions, we have the following decomposition on the in-sample prediction error (*e.g.*, Wahba, 1990):

$$\begin{aligned}
\frac{1}{n}\,\mathbb{E}_\epsilon\,\|\hat{y} - y^*\|^2 &= \frac{1}{n}\,\mathbb{E}_\epsilon\,\|K(K + n\lambda I)^{-1}(y^* + \epsilon) - y^*\|^2 \\
&= \frac{1}{n}\,\mathbb{E}_\epsilon\,\|(K(K + n\lambda I)^{-1} - I)y^* + K(K + n\lambda I)^{-1}\epsilon\|^2 \\
&= \frac{1}{n}\|(I - K(K + n\lambda I)^{-1})y^*\|^2 + \frac{1}{n}\,\mathbb{E}_\epsilon\,\|K(K + n\lambda I)^{-1}\epsilon\|^2 \\
&= \underbrace{n\lambda^2\|(K + n\lambda I)^{-1}y^*\|^2}_{\text{bias}} + \underbrace{\frac{\sigma^2}{n}\operatorname{Tr}(K^2(K + n\lambda I)^{-2})}_{\text{variance}}.
\end{aligned}$$

Here, the bias term increases with a non-zero λ, depending on how well y^* decomposes on the eigenvectors of K, while the variance term decreases with λ, based on a quantity known as *degrees of freedom* (Hastie et al., 2009), given by $\mathrm{df}(\lambda) = \operatorname{Tr}(K^2(K + n\lambda I)^{-2})$. This quantity provides a notion of *effective dimension* for a given value of λ (indeed, it replaces the dimension in the analysis of a standard least-squares estimate). If $\hat{\mu}_k$ denote the eigenvalues of the normalized kernel matrix $\frac{1}{n}K$, we have

$$\mathrm{df}(\lambda) = \sum_i \left(\frac{\hat{\mu}_i}{\hat{\mu}_i + \lambda}\right)^2. \tag{1.19}$$

This decreases with λ at a rate which depends on the rate of decay of $\hat{\mu}_i$ (assuming they are arranged in decreasing order), and precise rates of convergence can be established by trading-off the two above terms when f^* is assumed to belong to various non-parametric classes.

In the *random design* setting where $x_{1:n}$ are no longer fixed but drawn randomly from a data distribution, the analysis is more involved but similar quantities appear (*e.g.*, Hsu et al., 2014). In this setting, kernel methods are often analyzed under assumptions that are based on the integral operator as defined in (1.16), but where ν is now the marginal of the data distribution on the input data. In particular, when its eigenvalues μ_i decay as $O(i^{-\alpha})$, $\alpha > 1$ (known as *capacity condition*, characterizing the "size" of the RKHS), then one may show excess risk bounds of order $O(n^{-\alpha/(\alpha+1)})$ when $f^* \in \mathcal{H}$, which interpolate between the slow $O(1/\sqrt{n})$ rate when $\alpha \approx 1$ and the fast $O(1/n)$ rate when $\alpha \to \infty$ (*e.g.*, for exponential decays as is common for the Gaussian kernel). One can also obtain more refined rates when placing additional conditions (so-called *source conditions*) on the smoothness of f^*, even for $f^* \notin \mathcal{H}$, which allow a more precise control of approximation error. We refer, *e.g.*, to Caponnetto and De Vito (2007); Fischer and Steinwart (2017); Lin et al. (2018) for more details. A similar analysis may also be carried for other learning algorithms, such as early stopping (Yao et al., 2007) or stochastic gradient methods (Dieuleveut et al., 2016).

1.3.5 Speeding up kernel methods with kernel approximations

One drawback of kernel methods that arose with the surge of big datasets is their poor scalability to settings where n is very large. Indeed, while the kernel trick and representer theorem make kernel methods tractable, the algorithms presented in Section 1.3.3 quickly become computationally prohibitive with large n as they depend on the kernel matrix, and thus scale at least quadratically with n. Luckily, there has been work on providing good approximations of such a kernel matrix, making such approaches more scalable, while preserving good prediction properties. There are two main approaches in this direction: Nyström/column sampling, and random features.

Nyström or column sampling. This approach tries to exploit the fact that the kernel matrix is often approximately low rank, so that more efficient computations may be achieved through approximation. This is often achieved through selecting or sampling some columns (Smola and Schölkopf, 2000; Williams and Seeger, 2001) or through incomplete Cholesky decompositions (Fine and Scheinberg, 2001; Bach and Jordan, 2005). In a simple column sampling approach, one randomly selects p anchor points $z_1, \dots, z_p$ from the dataset,[3] and uses the following finite-dimensional subspace of the RKHS $\mathcal{H}$ instead of the full RKHS for learning:

$$\mathcal{F} = \operatorname{span}(\Phi(z_1), \dots, \Phi(z_p)).$$

Then, points $\Phi(x)$ can be approximated by their orthogonal projections on $\mathcal{F}$, denoted $\Pi\Phi(x)$, leading to inner product approximations of the form

$$\langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}} \approx \langle \Pi\Phi(x), \Pi\Phi(x') \rangle_{\mathcal{H}} = K_Z(x)^\top K_{ZZ}^{-1} K_Z(x'),$$

where we use the notation $K_Z(x) = (K(z_1, x), \dots, K(z_p, x))^\top$ and $(K_{ZZ})_{ij} = K(z_i, z_j)$. In particular, this can be written as $\langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}} \approx \langle \psi(x), \psi(x') \rangle_2$, where $\langle \cdot, \cdot \rangle_2$ is the Euclidian inner product, and $\psi(x) = K_{ZZ}^{-1/2} K_Z(x) \in \mathbb{R}^p$ can be used as finite-dimensional feature map, including at test time. Denoting $K_{ZX} \in \mathbb{R}^{p \times n}$ the matrix with entries $(K_{ZX})_{ij} = K(z_i, x_j)$, the corresponding kernel matrix is then given by

$$\tilde{K} = K_{ZX}^\top K_{ZZ}^{-1} K_{ZX},$$

which can be seen as a low rank approximation to the full matrix K.

It is then possible to study the influence of p, the number of sampled columns, on learning performance. While initial work has mainly analyzed approximation properties in matrix norms, more recent work has showed that by directly focusing on prediction performance on a given learning task, at least in the context of ridge regression, then p need only be of the order of degrees of freedom-like quantities similar to (1.19) in order to preserve good convergence rates (Bach, 2013; El Alaoui and Mahoney, 2015; Rudi et al., 2015). In particular, such quantities are often much smaller than n, leading to significant computational gains.

[3] We note that other strategies are possible for selecting anchor points, such as through clustering techniques, as in Zhang et al. (2008); Mairal (2016).

Random features. A second popular approach for improving the tractability of kernel methods is to leverage a favorable structure of some kernels which may be defined as expectations, that is, of the form

$$K(x, x') = \mathbb{E}_{w \sim p(w)}[\varphi(x; w)\varphi(x'; w)],$$

where $\varphi(x; w)$ are known as *random features* and $p(w)$ is a probability measure. Such a structure appears naturally for translation-invariant kernels $K(x, x') = \kappa(x - x')$, where Bochner's theorem yields $p(w) \approx \hat{\kappa}$, and real-valued random features $\varphi(x; w)$ can be constructed, known as random Fourier features (Rahimi and Recht, 2007). Other such kernels appear when considering infinitely-wide neural networks with one hidden layer, where the random features are now given by $\varphi(x; w) = \sigma(w^\top x)$, where σ is an activation function and one may optionally add a (random) bias term (Cho and Saul, 2009; Bach, 2017a; Daniely et al., 2016).

Learning with random features consists of sampling $w_1, \dots, w_m$ i.i.d from $p(w)$ and using a finite-dimensional feature map $\psi(x) = \frac{1}{\sqrt{m}}(\varphi(x; w_1), \dots, \varphi(x; w_m))^\top$, so that when m is large we have

$$K(x, x') \approx \langle \psi(x), \psi(x') \rangle_2, \tag{1.20}$$

and $\psi(\cdot)$ may thus be used as a finite-dimensional feature map for learning, leading to computational benefits when m is not too large.

It is possible to control the deviation in the approximation (1.20) with high probability over a compact domain (Rahimi and Recht, 2007), however this may only yield poor generalization bounds unless the number of random features m is larger than n, which would lose the computational benefits of approximation compared to using the full kernel. Yet, by using more refined analyses that directly tackle generalization rather than kernel approximation, one can show as in the Nyström case that a number of features of the order of a degrees of freedom quantity may be sufficient for good prediction performance (Bach, 2017b; Rudi and Rosasco, 2017). Rudi and Rosasco (2017) also show that for ridge regression, $\tilde{O}(\sqrt{n})$ random features are sufficient for obtaining worst-case $O(1/\sqrt{n})$ generalization bounds.

A point worth mentioning is that in contrast to the Nyström approach above, the models obtained via random features do not generally belong to the original RKHS, and therefore may lack desired smoothness properties.

1.4 Kernels for (Deep) Neural Networks

This section focuses on relationships between kernel methods and (deep) neural networks, such as kernels described by two-layer networks with an infinitely-wide hidden layer, and hierarchical constructions of kernels which can lead to descriptions of deep architectures.

1.4.1 Kernels from two-layer networks

The starting point that establishes links between kernels and deep networks is the early work on the Gaussian process behavior of infinitely wide networks with one hidden layer (Neal, 1996; Williams, 1997). Considering a neural network of the form

$$f(x) = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} v_i \sigma(w_i^\top x), \tag{1.21}$$

where $v_i \in \mathbb{R}$ and $w_i \in \mathbb{R}^p$ for $i = 1, \dots, m$ are parameters and σ an activation function such as the rectified linear unit (ReLU), $\sigma(u) = \max(u, 0)$. If we let $m \to \infty$ with i.i.d. parameters $v_i \sim \mathcal{N}(0, 1)$ and $w_i \sim p(w)$ for some probability measure $p(w)$, we obtain that f tends to a centered Gaussian process with covariance

$$K(x, x') = \lim_{m \to \infty} \mathbb{E}[f(x)f(x')] = \mathbb{E}_{w \sim p(w)}[\sigma(w^\top x)\sigma(w^\top x')],$$

which corresponds precisely to a random features kernel as described in Section 1.3.5, and motivates the use of such kernels to approximate the inductive bias of wide two-layer neural networks through a Gaussian process prior, or in general kernel methods.

Explicit forms of the kernels. For the example of the ReLU activation with $w \sim \mathcal{N}(0, 2I)$, the kernel can be expressed in closed form (Cho and Saul, 2009):

$$K(x, x') = \|x\|\|x'\|\kappa\left(\frac{x^\top x'}{\|x\|\|x'\|}\right), \quad \text{with} \quad \kappa(u) = \frac{1}{\pi}(u \cdot (\pi - \arccos(u)) + \sqrt{1 - u^2}). \tag{1.22}$$

Cho and Saul (2009) similarly characterize kernels for other positively homogeneous activations of the form $\sigma_\alpha(u) = \max(u^\alpha, 0)$, for $\alpha \in \mathbb{N}$. When inputs are restricted to the p-dimensional sphere $\mathcal{X} = \mathbb{S}^{p-1}$ and $w \sim \mathcal{N}(0, I)$, Daniely et al. (2016) show how to obtain explicit expressions of K in terms of the decomposition of the activation function σ in the basis of Hermite polynomials.[4] Specifically, if $(a_i)_i$ are the coefficients of the expansion of σ in the Hermite basis, then we have, for $x, x' \in \mathbb{S}^{p-1}$,

$$K(x, x') = \kappa(x^\top x'), \quad \text{with} \quad \kappa(u) = \sum_i a_i^2 u^i, \tag{1.23}$$

where κ is sometimes called *dual activation*. This result leads to explicit expressions for a few choices of σ in addition to positively homogeneous activations, and provides calculus rules to compute such dual activations; see Daniely et al. (2016) for more details.

Description of the RKHS. As shown in Bach (2017a), the RKHS corresponding to kernel K, denoted $\mathcal{H}$, contains functions of the form

$$f(x) = \int h(w)\sigma(w^\top x)dp(w),$$

with $h \in L^2(dp)$, with norm $\|f\|_{\mathcal{H}} = \inf\{\|h'\|_{L^2(dp)},\ h' \text{ s.t. } f(x) = \int h'(w)\sigma(w^\top x)dp(w)\}$ (this is a consequence of Theorem 1.3). Such a function f takes the form of an infinitely wide neural network, with output weights given by $h(w)$ for a given hidden neuron parameterized by w. The random features approximation with m features then yields a finite-width neural network, where the first layer weights w are fixed and equal to their random initialization, while the output weights are learned with an ℓ^2 penalty. When the number m of such neurons is large enough, one can still learn nearly as well as for infinite width, as discussed in Section 1.3.5 for random features.

For positively homogeneous activations σ_α, Bach (2017a) studies their decomposition in the basis of spherical harmonics when inputs are restricted to the p-dimensional

[4] We recall that Hermite polynomials define an orthonormal basis of $L^2(\mathbb{R})$ equipped with the standard Gaussian measure.

sphere $\mathcal{X} = \mathbb{S}^{p-1}$, which leads to Mercer decompositions of the corresponding kernel, and hence a precise characterization of the RKHS. By studying the eigenvalue decay of such decompositions, this leads to necessary regularity conditions for belonging to $\mathcal{H}$, as well as rates of approximations to Lipschitz functions defined on the sphere. For activations σ_α, the eigenvalues of the kernel corresponding to spherical harmonics of degree k are of order $O(k^{-p-2\alpha})$, leading to regularity requirements on derivatives up to order $p/2 + \alpha$. This is discussed in more detail in Chapter 4.

It is important to mention that while such kernels provide good approximation properties on the sphere, the RKHS $\mathcal{H}$ does not necessarily contain simple neural network functions, such as a single basis function $x \mapsto \sigma(w_*^\top x)$. This is in contrast to a learning procedure which might adapt the position of the neurons in the first layer depending on the data, such as learning with an ℓ^1 penalty on the second layer weights (or rather, a total-variation norm on the corresponding measure), which may however be intractable (see Bach, 2017a).

Dimension-free RKHS description. The description above relies on spherical harmonic decompositions, which depend on the underlying dimension p. A less precise description may be obtained independently of the dimension by using the decomposition (1.23), assuming that inputs lie on the sphere. Indeed, we can write $K(x, x') = \langle \phi(x), \phi(x') \rangle_2$, with the explicit feature map

$$\phi(x) = (a_0, a_1 x, a_2 x \otimes x, \dots, a_k x^{\otimes k}, \dots), \tag{1.24}$$

where $x^{\otimes k} = x \otimes \cdots \otimes x$ (k times) denotes the k-th order tensor product, and is such that $(x^{\otimes k})_{i_1,\dots,i_k} = x_{i_1} \cdots x_{i_k}$. Note that such a decomposition also holds for other dot-product kernels which may not directly come from a known activation, such as the inverse-polynomial kernel:

$$K(x, x') = \frac{1}{2 - \langle x, x' \rangle},$$

for which $a_i^2 = 2^{-(i+1)}$ (see Shalev-Shwartz et al., 2011; Zhang et al., 2016, 2017b). By Theorem 1.3, the RKHS then contains functions of the form $f_v(x) = \langle v, \phi(x) \rangle_2$, with norm $\|f_v\|_{\mathcal{H}} = \inf\{\|v'\|_2 : f_v = f_{v'}\}$. Unlike the Mercer decomposition above, this identity does not readily provide closed-form expressions of the norm, since the explicit feature map does not correspond to an orthogonal basis, but it still provides an upper bound $\|f_v\|_{\mathcal{H}} \leq \|v\|_2$.[5]

Application to improper learning. The approach we just described has been used to show that a certain class of neural networks with smooth activations is in the RKHS of specific kernels with a known upper bound on the norm, and thus are *improperly learnable* using kernel methods with this kernel. This means that learning with such kernels is guaranteed to perform at least as well as learning with this class (*e.g.*, Zhang et al., 2016, 2017b). Results of a similar flavor have been shown in the context of learning with gradient descent on over-parameterized neural networks, where the learning process

[5] We note that such a decomposition may still provide precise characterizations asymptotically in high dimensions, since Hermite and Legendre decompositions coincide for $p \to \infty$; see, *e.g.*, Ghorbani et al. (2019).

may approximate a kernel method with a certain kernel, and hence can *improperly* learn functions in the RKHS (*e.g.*, Allen-Zhu et al., 2019a; Arora et al., 2019b). In particular, one may consider functions of the form

$$g_w(x) = \sum_{i \geq 0} \gamma_i (w^\top x)^i = \theta(w^\top x), \tag{1.25}$$

where $\theta(u) = \sum_{i \geq 0} \gamma_i u^i$, which may be seen as a neural network function with a smooth activation function θ. Then, we can write $g_w(x) = \langle v_w, \phi(x) \rangle_2$ with

$$v_w = \left(\frac{\gamma_i}{a_i} w^{\otimes i} \right)_{i \geq 0},$$

assuming $\gamma_i = 0$ whenever $a_i = 0$. We then have $g_w \in \mathcal{H}$ if $\|v_w\|_2 < \infty$, with $\|g_w\|_{\mathcal{H}} \leq \|v_w\|_2$. This last quantity is often written as $C_\theta(\|w\|^2)$, with

$$C_\theta(B^2) := \sqrt{\sum_{i \geq 0} \left(\frac{\gamma_i}{a_i} \right)^2 B^{2i}}.$$

The quantity $C(\|w\|^2)$ may be seen as a measure of *complexity* of the corresponding neural network, and can be used to upper bound the Rademacher complexity of such a class of neural networks with smooth activations and constraints on weights. This approach can be extended to deep networks, through the use of hierarchical kernels, as we present next.

1.4.2 Hierarchical kernels

In many successful applications of neural networks, the architectures used involve multiple layers, raising the question of how to extend the approach of the previous section to include a form of depth or hierarchy.

Kernel construction. A reasonable approach is to define a kernel in terms of the geometry induced by another kernel. For instance, given an initial Gaussian kernel $K(x, x') = e^{-\|x - x'\|^2 / 2\sigma^2}$ with feature map $\Phi : \mathcal{X} \to \mathcal{H}$ to its RKHS, one may define a hierarchical Gaussian kernel by

$$K_2(x, x') = e^{-\frac{\|\Phi(x) - \Phi(x')\|_{\mathcal{H}}^2}{2\sigma^2}} = e^{-\frac{K(x,x) + K(x',x') - 2K(x,x')}{2\sigma^2}} = e^{-\frac{1 - K(x,x')}{\sigma^2}}.$$

With an abuse of notation, this kernel may be written through an explicit feature map as

$$K_2(x, x') = \langle \Phi(\Phi(x)), \Phi(\Phi(x')) \rangle,$$

although technically, the outer Φ is different than the inner one, since it corresponds to a feature map for a Gaussian kernel with a different input space $\mathcal{X}' = \mathcal{H}$, making it more difficult to study theoretically. Yet, the hierarchical kernel K_2 defines a new p.d. kernel on $\mathcal{X}$ which can be studied on its own. An example which is perhaps easier to parse is the following: if we combine a Gaussian kernel with scale σ into a simple polynomial kernel $(x, x') \mapsto (x^\top x')^2$, then we obtain a new Gaussian kernel with smaller scale $\sigma / \sqrt{2}$.

While this example is a bit dull in itself (indeed, why not consider the smaller scale right away?), it suggests that combining kernels hierarchically may produce richer kernels with better approximation properties (in this case, a smaller scale leads to a slower eigenvalue decay as per Section 1.3.2, thus a "larger" RKHS).

The dot-product kernels obtained in section 1.4.1 are particularly well-suited for such a hierarchical construction. Indeed, considering inputs on the sphere and a kernel $K(x, x') = \kappa(x^\top x')$, we may easily define a hierarchical version of it by composing κ with itself:

$$K_L(x, x') = \kappa^{(L)}(x^\top x') = \underbrace{\kappa \circ \cdots \circ \kappa}_{L \text{ times}}(x^\top x').$$

A similar relation holds for *homogeneous* dot-product kernels $K(x, x') = \|x\|\|x'\|\kappa(\frac{x^\top x'}{\|x\|\|x'\|})$ satisfying $\kappa(1) = 1$, such as the kernel in (1.22). Now, the L-layer kernel is given by

$$K_L(x, x') = \|x\|\|x'\|\kappa^{(L)}\left(\frac{x^\top x'}{\|x\|\|x'\|}\right).$$

Such hierarchical kernels were introduced by Cho and Saul (2009) as a multi-layer extension of the kernel (1.22), and similar concepts were used for describing other architectures (Mairal et al., 2014; Mairal, 2016; Anselmi et al., 2015; Daniely et al., 2016). These kernels also arise as limit objects when considering infinitely-wide fully-connected neural networks, such as when training only the last layer at initialization (Daniely et al., 2016; Daniely, 2017), or as the covariance function when considering Gaussian process limits at infinite width (Lee et al., 2018; Matthews et al., 2018). Daniely et al. (2016) shows that for such constructions when $\kappa(1) = 1$, the kernel K_L tends to a degenerate limit when $L \to \infty$, suggesting that it may not be beneficial to have too many fully-connected layers in a row.

Networks in the RKHS. For a hierarchical kernel $K(x, x') = \kappa(K_0(x, x'))$, we may consider a similar decomposition to (1.23) based on the polynomial expansion of κ:

$$K(x, x') = \sum_i a_i^2 (K_0(x, x'))^i.$$

Then, if $\Phi_0 : \mathcal{X} \to \mathcal{H}_0$ is the canonical feature map of K_0, we may then define an explicit feature map for K similar to (1.24), given by

$$\psi(x) = (a_0, a_1\Phi_0(x), \ldots, a_k\Phi_0(x)^{\otimes k}, \ldots).$$

Following a similar construction to the non-hierarchical case, we may then construct functions of the form $f(x) = \theta(\langle g, \Phi_0(x)\rangle_{\mathcal{H}_0}) = \theta(g(x))$, for a smooth non-linearity θ as before and where $g \in \mathcal{H}_0$. We then have

$$\|f\|_{\mathcal{H}} \leq C_\theta(\|g\|_{\mathcal{H}_0}^2).$$

If $K_0(x, x')$ is defined in a similar way as $\kappa(K_1(x, x'))$, we may then consider g as a neural network function itself, or a linear combination thereof, such as

$$g(x) = \sum_{j=1}^m w_j \theta(\langle h_j, \Phi_1(x)\rangle_{\mathcal{H}_1}),$$

where $\Phi_1 : \mathcal{X} \to \mathcal{H}_1$ is the canonical feature map for K_1, $h_1, \dots, h_m \in \mathcal{H}_1$ and $w_1, \dots, w_m \in \mathbb{R}$. Iterating this process, we may obtain a deep neural network as an element of the RKHS of a deep hierarchical kernel, and we may bound its RKHS norm using compositions of the complexity function C_θ. Following Chapter 2, it is possible to show an upper bound on the norm of a neural network $f(x) = w_L^\top \theta(W_{L-1}\theta(\cdots W_2\theta(W_1 x)\cdots))$, of the form

$$\|f\|^2 \leq \|w_L\|^2 C_\theta^2(\|W_{L-1}\|_2^2 C_\theta^2(\cdots \|W_2\|_2^2 C_\theta^2(\|W_1\|_F^2)\cdots)),$$

where $\|\cdot\|_2$ and $\|\cdot\|_F$ denote the spectral norm and Frobenius norm of a matrix, respectively. Zhang et al. (2016) provide different upper bounds on such networks with ℓ^1 constraints on the weights.

1.4.3 (Deep) convolutional kernels

The above construction of hierarchical kernels, while providing a nice functional space for defining multi-layer fully-connected networks, merely describes a new kernel defined on input vectors, and does not seem to clearly reflect benefits of the hierarchy. In contrast, deep learning architectures often incorporate elements which can exploit a certain structure in the data. A popular example, which will be a key topic in later chapters, is the convolutional architecture. Such architectures attempt to exploit the local stationarity in natural signals (*i.e.*, localized features present in similar form across the spatial domain), as well as shift-invariance and their multi-scale nature.

Similar design principles can be encoded into a (possibly hierarchical) kernel, by following similar architectural choices as convolutional networks. Such kernels were studied both for shallow architectures (Bo et al., 2010; Raj et al., 2017) and deep ones (Bo et al., 2011; Mairal et al., 2014; Mairal, 2016; Daniely et al., 2016).

Using notations from later chapters, we may consider discrete input signals $x[u]$ in $\ell^2(\Omega, \mathbb{R}^p)$, meaning that the signal takes values $x[u] \in \mathbb{R}^p$ at location $u \in \Omega \subset \mathbb{Z}^d$ (we have, *e.g.*, $d = 2$ and $p = 3$ for RGB images), and $\sum_{u\in\Omega} \|x[u]\|^2 < \infty$. Rather than feeding the entire signal into a kernel mapping, we first consider local neighborhoods, known as *patches*, and feed each of them into a kernel mapping, so that the kernel performs comparisons at a patch level rather than on the entire signal, allowing to take into account local stationarity. Define $S \in \mathbb{Z}^d$ a small patch shape (*e.g.*, $S = \{-s, \dots, s\}^d$) and $P : \ell^2(\Omega, \mathbb{R}^p) \to \ell^2(\Omega, \mathbb{R}^{p|S|})$ the linear *patch extraction* operator on signals given by

$$Px[u] = (x[u + v])_{v\in S}.$$

Then, given a kernel k_1 defined on $\mathbb{R}^{p|S|}$ (a *patch* kernel) with RKHS $\mathcal{H}_1$ and feature map φ_1, we may define a convolutional kernel by

$$K(x, x') = \sum_{u\in\Omega} k(Px[u], Px'[u]),$$

where the sum may be restricted to points u such that $u + S \subset \Omega$, when Ω is finite, or zero-padding may be applied to the signal instead. This may seen seen as defining a feature map $\phi(x) \in \ell^2(\Omega, \mathcal{H}_1)$ given by

$$\phi(x)[u] = \varphi_1(Px[u]).$$

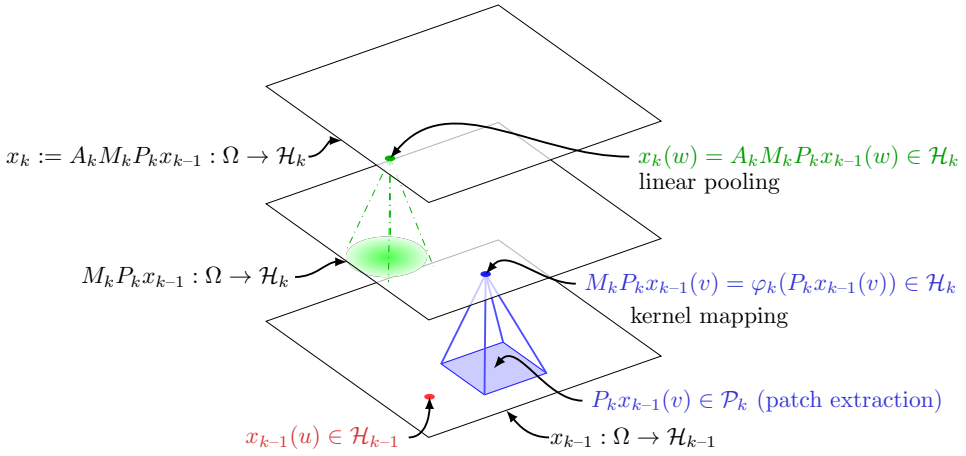


Figure 1.3: Construction of one layer of a convolutional kernel representation.

The kernel is then given by $K(x, x') = \langle \phi(x), \phi(x') \rangle_{\ell^2(\Omega, \mathcal{H}_1)}$.

One may then incorporate some shift-invariance through a pooling operation, either globally on the entire signal, or locally around each location. The former may be achieved through a local averaging of the feature map, *e.g.*, with a Gaussian or average pooling filter, through a linear convolution operator A given by

$$(A\phi(x))[u] = \sum_{v \in \Omega} h[u - v] \cdot \phi(x)[v],$$

where $h[u]$ is a given filter. This leads to a kernel of the form

$$\begin{aligned} K(x, x') &= \sum_u \sum_v \sum_{v'} h[u - v] h[u - v'] k(Px[v], Px'[v']) \\ &= \sum_v \sum_{v'} (h * h)[v - v'] k(Px[v], Px'[v']), \end{aligned}$$

where the last equality holds at least if we assume symmetric filters h and $\Omega = \mathbb{Z}^d$. This takes the form of a *match kernel* (*e.g.*, Bo et al., 2010; Mairal et al., 2014; Raj et al., 2017), where similarities between patches are weighted by a function of their distance. Using global pooling instead of local pooling simply yields the kernel

$$K(x, x') = \sum_v \sum_{v'} k(Px[v], Px'[v']). \tag{1.26}$$

An illustration for such a one-layer kernel representation consisting of patch extraction, patch kernel mapping, and local pooling is given in Figure 1.3. This construction may be extended in a multi-layer fashion, by considering further patch extraction, patch kernels and pooling on top of the obtained feature map $A\phi(x)[u]$. Such a construction is provided in detail in Chapter 2, where the corresponding RKHS is also studied. In particular, one can construct generic CNNs with smooth activations in the RKHS, and provide upper bounds on their RKHS norm, in a similar fashion to the fully-connected case presented in the previous section.

1.4.4 Neural tangent kernels

The hierarchical kernels described in the previous sections arise naturally in overparameterized networks (that is, with large or infinite width), when considering their Gaussian process behavior (Lee et al., 2018; Matthews et al., 2018), or when training only the last layer and keeping other layers fixed at random initialization (Daniely et al., 2017). A different line of work has shown that such over-parameterized networks are easy to optimize with gradient descent, and may converge in the infinite-width limit to a minimum-norm solution in the RKHS corresponding to a different kernel known as *neural tangent kernel* (NTK, Jacot et al., 2018). See, *e.g.*, Allen-Zhu et al. (2019b); Chizat et al. (2019); Du et al. (2019b,a); Li and Liang (2018). This kernel is given by the infinite-width limit corresponding to a feature map given by the gradient of the model with respect to its parameters, at initialization. For instance, for a two-layer network of the form (1.21), the NTK is of the form

$$\begin{aligned} K_{\text{ntk}}(x, x') &= \lim_{m \to \infty} \langle \nabla_{v,w} f(x), \nabla_{v,w} f(x') \rangle \\ &= (x^\top x') \, \mathbb{E}_{w \sim p(w)}[\sigma'(w^\top x)\sigma'(w^\top x')] + \mathbb{E}_{w \sim p(w)}[\sigma(w^\top x)\sigma(w^\top x')]. \end{aligned}$$

Such kernels are studied in more detail in the contribution (Bietti and Mairal, 2019b) of this thesis, detailed in Chapter 4, where we derive the NTK for a class of convolutional architectures, and compare its properties to the more straightforward hierarchical kernels from previous sections.

1.5 Invariance and Stability to Deformations

Many learning problems may benefit from prior knowledge expressed in terms of invariance to certain transformations. An example is translation-invariance for images in a classification task; indeed, one may expect that two images that are identical up to a translation should lead to the same prediction, making it useful to provide such an inductive bias in the choice of representation or function class, for improved sample complexity. Beyond translations, one may be interested in different groups of transformations, such as rigid transformations, or more local transformations on signals such as deformations.

1.5.1 Group invariant representations

Consider a group G of transformations to which we would like to be invariant, such as the translation or rotation group, where we denote by $L_g x$ the transformation of an example x by a given group element $g \in G$. For instance, if x is a signal $x(u)$ on a continuous domain, then we may consider

$$L_g x(u) = x(g^{-1} \cdot u),$$

where $g^{-1} \cdot u$ denotes the action of g^{-1} on u. For translations, this may simply be $L_g x(u) = x(u - g)$.

Invariance via averaging. One way to make a data representation Φ invariant to the group G is to consider an averaged representation over the group. Assuming the group is locally compact, we can consider the right-invariant Haar measure μ, which satisfies $\mu(Sg) = \mu(S)$ for any $S \subseteq G$ and $g \in G$. Then, we may define the averaged representation

$$\bar{\Phi}(x) = \int_G \Phi(L_{g'}x)d\mu(g').$$

We then have that $\bar{\Phi}$ is an invariant representation:

$$\bar{\Phi}(L_g x) = \int_G \Phi(L_{g'}L_g x)d\mu(g') = \int_G \Phi(L_{g'g}x)d\mu(g') = \int_G \Phi(L_{g'}x)d\mu(g') = \bar{\Phi}(x).$$

If Φ is a kernel mapping for a kernel K, then this corresponds to learning with a different kernel of the form

$$\bar{K}(x, x') = \int\int K(L_g x, L_{g'}x')d\mu(g)d\mu(g').$$

The convolutional kernel with global average pooling in (1.26) is an example of such a kernel, and is this invariant to translations assuming the average is over all translations. Using a weighted average instead of such a global average may lead to a more *local* form of invariance. See Anselmi et al. (2016); Mroueh et al. (2015); Raj et al. (2017) for further details on such invariant kernels. Other forms of pooling based on hierarchical maxima can also yield invariance and information preservation properties (Smale et al., 2010; Bouvrie et al., 2009).

Hierarchy and equivariance. For signal representations that are defined hierarchically in a multi-layer fashion, it can be useful to preserve a given group structure across layers. For instance, for signals defined on a domain $\Omega = \mathbb{R}^d$, there is an inherent structure given by the translation group, and one may obtain new feature maps at each layers defined themselves on the same domain. A desirable property of such hierarchical constructions is that the obtained feature maps are *equivariant* (or *covariant*) to the group action, meaning that the feature maps obtained from a transformed signal are equal to the corresponding transformation of the feature map obtained on the original signal. Formally, a representation $\Phi(x)$ of a signal x is equivariant to the action of a group G if for any $g \in G$ we have

$$\Phi(L_g x) = L_g \Phi(x).$$

Here, both x and $\Phi(x)$ are defined on the group G, or on subsets thereof (such as quotient spaces of the group, see Kondor and Trivedi (2018)). Convolutional layers in deep networks have this property for the translation group, as do the patch extraction operators used in convolutional kernels. One may define similar operations to convolutions for other groups, obtaining multi-layer networks or kernels with desirable equivariance properties for other transformations such as rotations, reflections, or roto-translations (which combines translations and rotations); see, *e.g.*, Cohen and Welling (2016); Cohen et al. (2018); Kondor and Trivedi (2018); Oyallon and Mallat (2015); Sifre and Mallat (2013) and Chapter 2. Note that such ideas can be extended to signals defined on domains that may lack Euclidian or group structures, such as graphs or manifolds; see Bronstein et al. (2017) for a review.

Figure 1.4: Examples of variabilities in handwritten digits which may be seen as small deformations and should lead to the same predictions in a classification task: (top) variability due to different hand-writings; (bottom) small, hand-crafted deformations of a given image of the digit 5.

While equivariance in itself does not provide invariance, one can easily obtain an invariant representation from an equivariant one by averaging the obtained representation on the group:

$$\bar{\Phi}(x) = \int_G L_g \Phi(x) d\mu(g),$$

which, by the equivariance property, is equivalent to averaging on transformed input signals. Such an equivariant construction can help achieve invariance jointly with other desirable properties such as stability to deformations, by only averaging on a specific group in the final representation. Such constructions were initially considered in the context of the scattering transform by Mallat (2012); Sifre and Mallat (2013); Oyallon and Mallat (2015), and in Chapter 2 of this thesis, we analyze such joint stability and invariance properties for the roto-translation group.

1.5.2 Stability to deformations

While invariance to groups of rigid transformations such as translations is clearly helpful for learning from natural signals, such data typically presents much richer classes of variabilities to which we would like to be nearly invariant. An example is to consider small local perturbations around translations, given by deformation fields, as pictured in Figure 1.4 for hand-written digits. While one may be able to deform an image of the digit 1 into a 7, we may expect that when the deformation is "small", the label of the image should be preserved. The stability of signal representations to such smooth deformations can be studied mathematically, a line of work initiated by Mallat (2012) with the *scattering transform*, a wavelet-based representation of signals which has been successful for images, textures, and audio signals among others (Bruna and Mallat, 2013; Oyallon et al., 2017; Sifre and Mallat, 2013; Andén and Mallat, 2014). It is an important contribution of this thesis to extend such a mathematical study to representations given by more standard convolutional architectures, and to relate such stability properties to learning guarantees through kernel methods; see Chapter 2 and the corresponding publications (Bietti and Mairal, 2017a, 2019a).

Motivation. Representations that are globally invariant to translations are important, yet some simple approaches to obtain such representations may lack some other

desirable properties. For instance, a global averaging of a local representation taken on small patches may throw away information about interactions at larger scales, which can provide useful discriminative features for learning.

Another example of a simple translation-invariant representation is the modulus of the Fourier transform, given by $\Phi(x) = |\hat{x}|$ for a signal $x(u)$, where $\hat{x}(\omega)$ is the Fourier transform of x. Such a representation is translation-invariant, since a translation only affects the phase of the Fourier transform, and thus does not change its modulus. It also preserves much of the signal information, as in many tasks most of the useful information is in the amplitude of the Fourier transform. Yet, it can be shown that such a representation is unstable to small deformations, such as a small dilation $x_\epsilon(u) = x((1-\epsilon)u)$ with a small $\epsilon > 0$; indeed, $||\hat{x}| - |\hat{x}_\epsilon||$ can be arbitrarily large if x has isolated high frequencies (see, *e.g.*, Mallat, 2012; Bruna and Mallat, 2013). Such instabilities can be mitigated through local averaging in frequency domain, with a bandwidth that becomes larger for high frequencies as in mel-frequency spectrograms, which behaves similarly to certain wavelet filter banks followed by averaging in order to obtain some invariance (see Andén and Mallat, 2014). Nevertheless, such averaging operations may lead to information loss.

These examples suggest that more elaborate representations, possibly defined hierarchically, may be needed to obtain deformation stability while preserving information about the original signal.

Definitions of stability. A deformation may be seen as a vector field of translations applied to each location u in a signal $x(u)$. For such a deformation $\tau : \Omega \to \Omega$, assumed to be a C^1 diffeomorphism, we denote a deformed image by $L_\tau x$, given by $L_\tau x(u) = x(u - \tau(u))$. Following the definitions of Mallat (2012), we will say that a signal representation Φ is stable to deformations if

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1 \|\nabla \tau\|_\infty + C_2 \|\tau\|_\infty)\|x\|,$$

where $\nabla\tau$ denotes the Jacobian of τ, and we define

$$\|\nabla\tau\|_\infty := \sup_{u \in \Omega} \|\nabla\tau(u)\|, \quad \text{and} \quad \|\tau\|_\infty := \sup_{u \in \Omega} |\tau(u)|.$$

Note that one may include further terms based on higher-order derivatives of τ; for instance, Mallat (2012) includes a term depending on

$$\|\nabla^2\tau\|_\infty := \sup_{u \in \Omega} \|\nabla^2\tau(u)\|,$$

where $\|\nabla^2\tau(u)\|$ denotes the operator norm of the Hessian tensor of τ at u. One typically assumes $\|\nabla\tau\|_\infty < 1$ to ensure invertibility of the deformation. If Φ is translation-invariant, then we have $C_2 = 0$, but we may consider near-translation invariant representations for which C_2 is non-zero but small. When τ deviates from a translation, $\nabla\tau$ is non-zero, but stability guarantees that the representation does not deviate too much when the operator norm of $\nabla\tau$ is small. For instance, a small scaling (*i.e.*, $u - \tau(u) = (1-\epsilon)u$) or small rotation (*i.e.*, $u - \tau(u) = R_\epsilon u$, with R_ϵ a rotation matrix with small angle) lead to $\|\nabla\tau\|_\infty = O(\epsilon)$.

Assumptions on the class of signals. Different assumptions on the signals may be needed in order to achieve deformation stability, depending on the choice of representation. The signal x is typically assumed to by in $L^2(\mathbb{R}^d)$, and some representations may additionally assume that the signal has little or no energy beyond a certain level of frequencies. Such an assumption on the signals frequencies is made in our work in Chapter 2 and in the work of Wiatowski and Bölcskei (2018), while Mallat (2012) does not require such an assumption, with stability guarantees which hold uniformly for any frequential support. This is made possible by the joint treatment of all frequencies in the wavelet transforms considered by Mallat (2012), while in Chapter 2 we treat each layer of a deep convolutional representation separately, corresponding to different scales.

Achieving stability. As discussed above, instabilities can arise when applying high-frequency filters on a signal with isolated high-frequencies. The *scattering representation* of Mallat (2012) overcomes this by applying cascades of well-chosen wavelet transforms and modulus non-linearities, while extracting coefficients through local averages. Such wavelets are localized and can be shown to be stable to deformations, for any input signal x in $L^2(\mathbb{R}^d)$. They are equivariant to translations, thus other mechanisms are needed for invariance, in this case modulus non-linearities followed by average pooling at a target scale of invariance. Such pooled coefficients capture variabilities at certain scales, and the signal energy lost from pooling is recovered through the following layers, ensuring that the energy of the original signal is preserved for representations with enough layers. While signal energy only concerns the norm of the signal, recovery of the original signal may also be possible through a form of phase recovery.

Extending such properties to deep convolutional networks with learned filter is more complicated, as the filters may not have desirable properties for stability as wavelets do. Wiatowski and Bölcskei (2018) show quite general guarantees on stability for arbitrary convolutional networks on band-limited signals, that is, signals with no energy beyond a certain frequency R. However, such results are quite weak, essentially obtaining the same stability guarantee as the original signal itself, which grows linearly with the cutoff frequency R, and thus is not realistic for signals with high-frequencies, as R may be quite large in practice. In contrast, in Chapter 2 we show that for appropriate architectures with well-chosen patch sizes and pooling layers, stability can be achieved with only a logarithmic dependence on R. We also show that the representation given by multi-layer convolutional kernels with such architectures may preserve signal information by allowing reconstruction of the original signal.

1.6 Adversarial Robustness

A notion related to the invariance and stability of representations is the robustness of a machine learning model to small perturbations. In particular, it is desirable to learn models that not only generalize well to unseen samples, but to do so in a robust manner, meaning that small perturbations of the test examples will also result in predictions similar to those for the clean examples. Such issues are important to take into account for security concerns, and have been recently popularized through the observation that deep neural networks tend to be very sensitive to small, imperceptible perturbations known as *adversarial examples*; see Figure 1.5. In Chapter 3, we tackle the problem of

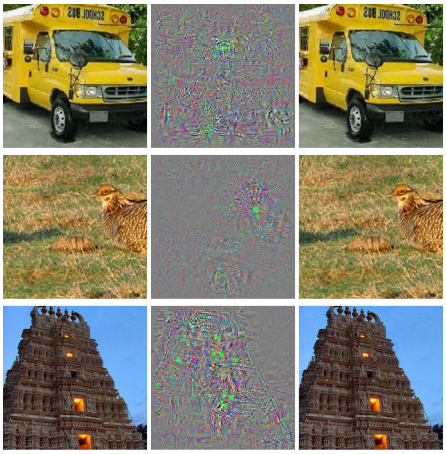


Figure 1.5: Illustration of adversarial examples on ImageNet images for an AlexNet deep network; from Szegedy et al. (2014). Clean images with correct predictions (left) can be modified with well-chosen, imperceptible perturbations (middle), so that the resulting images (right) fool the model to predict the label "ostrich".

adversarial robustness from the lens of regularization and kernel methods, and obtain guarantees on robust generalization as well as state-of-the-art performance for robustness to ℓ_2 perturbations on the CIFAR10 dataset.

Goal of robust learning. We may formulate a different goal for statistical learning of robust models, assuming a perturbation set denoted $B_\epsilon(x)$ around a point x. Typically, this is chosen to be a set of additive perturbations in an ϵ-ball around x, such as $B_\epsilon(x) = \{x + \delta, \|\delta\| \leq \epsilon\}$, for some norm $\|\cdot\|$, commonly taken to be the ℓ_2 or ℓ_∞ norm. Then, the goal is to find a good predictor in terms of expected *robust* loss:

$$L^\epsilon(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sup_{x'\in B_\epsilon(x)} \ell(f(x'), y)\right].$$

It is possible to show that robust supervised learning is a distinct goal from standard supervised learning, which attempts to minimize $L(f) = L^0(f)$. Indeed, there may be features that are slightly correlated with the label despite being non-robust with respect to a small perturbation, making them useful for standard supervised learning, but not for robust learning (see, *e.g.*, Tsipras et al., 2019, for an example). This suggests that different learning algorithms may be needed for robustness.

Approaches to robustness. The observations that neural networks are highly sensitive to adversarial perturbations have led researchers to propose various strategies to obtain robust models, which in the terminology of security are often referred to as "defenses" to adversarial "attacks". A key ingredient in many defense strategies is to train models by attempting to optimize the following empirical robust loss,

$$\hat{L}^\epsilon(f) = \frac{1}{n}\sum_{i=1}^{n} \sup_{x_i'\in B_\epsilon(x_i)} \ell(f(x_i'), y_i),$$

leading to a *robust optimization* problem. In the context of training deep networks with SGD, this is typically achieved by finding approximate maximizers for the perturbed examples x_i' in a batch using a few steps of (variants of) projected gradient descent, and then performing back-propagation on the model f at these points (see, *e.g.*, Madry et al., 2018). The inner loop which finds perturbed examples essentially boils down to finding adversarial examples that are then used for training, which is why variants of this method are sometimes referred to as *adversarial training*.

While the approach described above often lead to good empirical robustness performance against standard attacks at test time, they do not guarantee such a performance for all possible attacks. This has motivated various authors to obtain *certified* methods, meaning that one can guarantee a given performance for all possible attackers constrained to B_ϵ. This can be achieved by obtaining upper bound certificates on the robust optimization problem above, for instance by propagating the entire constraint set through the layers of the neural network (*e.g.*, Wong and Kolter, 2018) or through relaxations (Raghunathan et al., 2018). Another approach which was found to be more scalable is to randomize the classifier, for instance through smoothing by injecting noise, in order to provide high-probability certified guarantees (Lecuyer et al., 2019; Cohen et al., 2019; Salman et al., 2019).

Links with regularization. In the case of a linear or kernel-based model, the norm may provide a control of stability and robustness; indeed, for a linear model $f(x) = w^\top x$ we have

$$\sup_{x' \in B_\epsilon(x)} f(x') = w^\top x + \epsilon \|w\|_*,$$

where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$. For an RKHS function $f(x) = \langle f, \Phi(x) \rangle_{\mathcal{H}}$, we may obtain a similar control of robustness for perturbations in an RKHS ball:

$$\sup_{\|u\|_{\mathcal{H}} \leq \epsilon} \langle f, \Phi(x) + u \rangle_{\mathcal{H}} = f(x) + \epsilon \|f\|_{\mathcal{H}}, \tag{1.27}$$

and such perturbations often include additive perturbations in the input space, for instance when the kernel mapping is non-expansive, in which case the norm also controls robustness to additive perturbations in B_ϵ by (1.27). Indeed,

$$\sup_{x' \in B_\epsilon(x)} f(x') = \sup_{x' \in B_\epsilon(x)} \langle f, \Phi(x) + (\Phi(x') - \Phi(x)) \rangle_{\mathcal{H}} \leq \sup_{\|u\|_{\mathcal{H}} \leq \epsilon} \langle f, \Phi(x) + u \rangle_{\mathcal{H}}.$$

Note that aiming for small generalization error alone does not necessarily require a small norm; indeed, even in a setting where the regression function itself belongs to the class with small norm, one may obtain good generalization with large-norm models, such as those obtained by interpolating noisy labels (*e.g.*, Bartlett et al., 2019; Liang and Rakhlin, 2019), but such a large norm could cause poor robustness properties. Then, controlling this dual norm through *regularization* may help obtain more robust models. We use this point of view for providing new algorithms which achieve state-of-the-art results in ℓ_2 robustness for deep models in Chapter 3, by providing ways to explicitly regularize a deep network using (approximations of) an RKHS norm. In Chapter 3, we also show that the margin bounds presented in Section 1.2.4 can be extended to provide generalization guarantees for robust generalization.

Chapter 2

Invariance, Stability to Deformations, and Complexity of Deep Convolutional Representations

The success of deep convolutional architectures is often attributed in part to their ability to learn multiscale and invariant representations of natural signals. However, a precise study of these properties and how they affect learning guarantees is still missing. In this chapter, we consider deep convolutional representations of signals; we study their invariance to translations and to more general groups of transformations, their stability to the action of diffeomorphisms, and their ability to preserve signal information. This analysis is carried by introducing a multilayer kernel based on convolutional kernel networks and by studying the geometry induced by the kernel mapping. We then characterize the corresponding reproducing kernel Hilbert space (RKHS), showing that it contains a large class of convolutional neural networks with homogeneous activation functions. This analysis allows us to separate data representation from learning, and to provide a canonical measure of model complexity, the RKHS norm, which controls both stability and generalization of any learned model. In addition to models in the constructed RKHS, our stability analysis also applies to convolutional networks with generic activations such as rectified linear units, and we discuss its relationship with recent generalization bounds based on spectral norms.

This chapter is based on the following material:

A. Bietti and J. Mairal. Invariance and stability of deep convolutional representations. In *Advances in Neural Information Processing Systems (NIPS)*, 2017a

A. Bietti and J. Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *Journal of Machine Learning Research (JMLR)*, 20(25):1–49, 2019a

2.1 Introduction

The results achieved by deep neural networks for prediction tasks have been impressive in domains where data is structured and available in large amounts. In particular, convolutional neural networks (CNNs, LeCun et al., 1989) have shown to effectively leverage the local stationarity of natural images at multiple scales thanks to convolutional operations, while also providing some translation invariance through pooling operations. Yet, the exact nature of this invariance and the characteristics of functional spaces where convolutional neural networks live are poorly understood; overall, these models are sometimes seen as clever engineering black boxes that have been designed with a lot of insight collected since they were introduced.

Understanding the inductive bias of these models is nevertheless a fundamental question. For instance, a better grasp of the geometry induced by convolutional representations may bring new intuition about their success, and lead to improved measures of model complexity. In turn, the issue of regularization may be solved by providing ways to control the variations of prediction functions in a principled manner. One meaningful way to study such variations is to consider the stability of model predictions to naturally occuring changes of input signals, such as translations and deformations.

Small deformations of natural signals often preserve their main characteristics, such as class labels (*e.g.*, the same digit with different handwritings may correspond to the same images up to small deformations), and provide a much richer class of transformations than translations. The scattering transform (Mallat, 2012; Bruna and Mallat, 2013) is a recent attempt to characterize convolutional multilayer architectures based on wavelets. The theory provides an elegant characterization of invariance and stability properties of signals represented via the scattering operator, through a notion of Lipschitz stability to the action of diffeomorphisms. Nevertheless, these networks do not involve "learning" in the classical sense since the filters of the networks are pre-defined, and the resulting architecture differs significantly from the most used ones, which adapt filters to training data.

In this work, we study these theoretical properties for more standard convolutional architectures, from the point of view of positive definite kernels (Schölkopf and Smola, 2001). Specifically, we consider a functional space derived from a kernel for multi-dimensional signals that admits a multi-layer and convolutional structure based on the construction of convolutional kernel networks (CKNs) introduced by Mairal (2016); Mairal et al. (2014). The kernel representation follows standard convolutional architectures, with patch extraction, non-linear (kernel) mappings, and pooling operations. We show that our functional space contains a large class of CNNs with smooth homogeneous activation functions.

The main motivation for introducing a kernel framework is to study separately data representation and predictive models. On the one hand, we study the translation-invariance properties of the kernel representation and its stability to the action of diffeomorphisms, obtaining similar guarantees as the scattering transform (Mallat, 2012), while preserving signal information. When the kernel is appropriately designed, we also show how to obtain signal representations that are invariant to the action of any locally compact group of transformations, by modifying the construction of the kernel representation to become *equivariant* to the group action. On the other hand, we show that these stability results can be translated to predictive models by controlling their norm

in the functional space, or simply the norm of the last layer in the case of CKNs (Mairal, 2016). With our kernel framework, the RKHS norm also acts as a measure of model complexity, thus controlling both stability and generalization, so that stability may lead to improved sample complexity. Finally, our work suggests that explicitly regularizing CNNs with the RKHS norm (or approximations thereof) can help obtain more stable models, a more practical question which we study in Chapter 3.

2.1.1 Summary of Main Results

Our work characterizes properties of deep convolutional models along two main directions.

- The first goal is to study *representation* properties of such models, independently of training data. Given a deep convolutional architecture, we study signal preservation as well as invariance and stability properties.
- The second goal focuses on *learning* aspects, by studying the complexity of learned models based on our representation. In particular, our construction relies on kernel methods, allowing us to define a corresponding functional space (the RKHS). We show that this functional space contains a class of CNNs with smooth homogeneous activations, and study the complexity of such models by considering their RKHS norm. This directly leads to statements on the generalization of such models, as well as on the invariance and stability properties of their predictions.
- Finally, we show how some of our arguments extend to more traditional CNNs with generic and possibly non-smooth activations (such as ReLU or tanh).

Signal preservation, invariance and stability. We tackle this first goal by defining a deep convolutional representation based on hierarchical kernels. We show that the representation preserves signal information and guarantees near-invariance to translations and stability to deformations in the following sense, defined by Mallat (2012): for signals $x : \mathbb{R}^d \to \mathbb{R}^{p_0}$ defined on the continuous domain $\mathbb{R}^d$, we say that a representation $\Phi(x)$ is *stable* to the action of diffeomorphisms if

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1 \|\nabla \tau\|_\infty + C_2 \|\tau\|_\infty)\|x\|,$$

where $\tau : \mathbb{R}^d \to \mathbb{R}^d$ is a C^1-diffeomorphism, $L_\tau x(u) = x(u - \tau(u))$ its action operator, and the norms $\|\tau\|_\infty$ and $\|\nabla \tau\|_\infty$ characterize how large the translation and deformation components are, respectively (see Section 2.3 for formal definitions). The Jacobian $\nabla\tau$ quantifies the size of local deformations, so that the first term controls the stability of the representation. In the case of translations, the first term vanishes ($\nabla\tau = 0$), hence a small value of C_2 is desirable for translation invariance. We show that such signal preservation and stability properties are valid for the multilayer kernel representation Φ defined in Section 2.2 by repeated application of patch extraction, kernel mapping, and pooling operators:

- The representation can be discretized with no loss of information, by subsampling at each layer with a factor smaller than the patch size;

- The translation invariance is controlled by a factor $C_2 = C_2'/\sigma_n$, where σ_n represents the "resolution" of the last layer, and typically increases exponentially with depth;

- The deformation stability is controlled by a factor C_1 which increases as κ^{d+1}, where κ corresponds to the patch size at a given layer, that is, the size of the "receptive field" of a patch relative to the resolution of the previous layer.

These results suggest that a good way to obtain a stable representation that preserves signal information is to use the smallest possible patches at each layer (*e.g.*, 3x3 for images) and perform pooling and downsampling at a factor smaller than the patch size, with as many layers as needed in order to reach a desired level of translation invariance σ_n. We show in Section 2.3.3 that the same invariance and stability guarantees hold when using kernel approximations as in CKNs, at the cost of losing signal information.

In Section 2.3.5, we show how to go beyond the translation group, by constructing similar representations that are invariant to the action of locally compact groups. This is achieved by modifying patch extraction and pooling operators so that they commute with the group action operator (this is known as *equivariance*).

Model complexity. Our second goal is to analyze the complexity of deep convolutional models by studying the functional space defined by our kernel representation, showing that certain classes of CNNs are contained in this space, and characterizing their norm.

The multi-layer kernel representation defined in Section 2.2 is constructed by using kernel mappings defined on local signal patches at each scale, which replace the linear mapping followed by a non-linearity in standard convolutional networks. Inspired by Zhang et al. (2017b), we show in Section 2.4.1 that when these kernel mappings come from a class of dot-product kernels, the corresponding RKHS contains functions of the form

$$z \mapsto \|z\|\sigma(\langle g, z\rangle/\|z\|),$$

for certain types of smooth activation functions σ, where g and z live in a particular Hilbert space. These behave like simple neural network functions on patches, up to homogeneization. Note that if σ was allowed to be homogeneous, such as for rectified linear units $\sigma(\alpha) = \max(\alpha, 0)$, homogeneization would disappear. By considering multiple such functions at each layer, we construct a CNN in the RKHS of the full multi-layer kernel in Section 2.4.2. Denoting such a CNN by f_σ, we show that its RKHS norm can be bounded as

$$\|f_\sigma\|^2 \leq \|w_{n+1}\|^2 \; C_\sigma^2(\|W_n\|_2^2 \; C_\sigma^2(\|W_{n-1}\|_2^2 \ldots C_\sigma^2(\|W_2\|_2^2 \; C_\sigma^2(\|W_1\|_F^2)) \ldots)),$$

where W_k are convolutional filter parameters at layer k, w_{n+1} carries the parameters of a final linear fully connected layer, C_σ^2 is a function quantifying the complexity of the simple functions defined above depending on the choice of activation σ, and $\|W_k\|_2$, $\|W_k\|_F$ denote spectral and Frobenius norms, respectively, (see Section 2.4.2 for details). This norm can then control generalization aspects through classical margin bounds, as well as the invariance and stability of model predictions. Indeed, by using the reproducing

property $f(x) = \langle f, \Phi(x) \rangle$, this "linearization" lets us control stability properties of model predictions through $\|f\|$:

$$\text{for all signals } x \text{ and } x', \quad |f(x) - f(x')| \leq \|f\| \cdot \|\Phi(x) - \Phi(x')\|,$$

meaning that the prediction function f will inherit the stability of Φ when $\|f\|$ is small.

The case of standard CNNs with generic activations. When considering CNNs with generic, possibly non-smooth activations such as rectified linear units (ReLUs), the separation between a data-independent representation and a learned model is not always achievable in contrast to our kernel approach. In particular, the "representation" given by the last layer of a learned CNN is often considered by practitioners, but such a representation is data-dependent in that it is typically trained on a specific task and dataset, and does not preserve signal information.

Nevertheless, we obtain similar invariance and stability properties for the predictions of such models in Section 2.4.3, by considering a complexity measure given by the product of spectral norms of each linear convolutional mapping in a CNN. Unlike our study based on kernel methods, such results do not say anything about generalization; however, relevant generalization bounds based on similar quantities have been derived (though other quantities in addition to the product of spectral norms appear in the bounds, and these bounds do not directly apply to CNNs), *e.g.*, by Bartlett et al. (2017); Neyshabur et al. (2018), making the relationship between generalization and stability clear in this context as well.

2.1.2 Related Work

Our work relies on image representations introduced in the context of convolutional kernel networks (Mairal, 2016; Mairal et al., 2014), which yield a sequence of spatial maps similar to traditional CNNs, but where each point on the maps is possibly infinite-dimensional and lives in a reproducing kernel Hilbert space (RKHS). The extension to signals with d spatial dimensions is straightforward. Since computing the corresponding Gram matrix as in classical kernel machines is computationally impractical, CKNs provide an approximation scheme consisting of learning finite-dimensional subspaces of each RKHS's layer, where the data is projected. The resulting architecture of CKNs resembles traditional CNNs with a subspace learning interpretation and different unsupervised learning principles.

Another major source of inspiration is the study of group-invariance and stability to the action of diffeomorphisms of scattering networks (Mallat, 2012), which introduced the main formalism and several proof techniques that were keys to our results. Our main effort was to extend them to more general CNN architectures and to the kernel framework, allowing us to provide a clear relationship between stability properties of the representation and generalization of learned CNN models. We note that an extension of scattering networks results to more general convolutional networks was previously given by Wiatowski and Bölcskei (2018); however, their guarantees on deformations do not improve on the inherent stability properties of the considered signal, and their study does not consider learning or generalization, by treating a convolutional architecture with fixed weights as a feature extractor. In contrast, our stability analysis shows the

benefits of deep representations with a clear dependence on the choice of network architecture through the size of convolutional patches and pooling layers, and we study the implications for learned CNNs through notions of model complexity.

Invariance to groups of transformations was also studied for more classical convolutional neural networks from methodological and empirical points of view (Bruna et al., 2013; Cohen and Welling, 2016), and for shallow learned representations (Anselmi et al., 2016) or kernel methods (Haasdonk and Burkhardt, 2007; Mroueh et al., 2015; Raj et al., 2017). Our work provides a similar group-equivariant construction to (Cohen and Welling, 2016), while additionally relating it to stability. In particular, we show that in order to achieve group invariance, pooling on the group is only needed at the final layer, while deep architectures with pooling at multiple scales are mainly beneficial for stability. For the specific example of the roto-translation group (Sifre and Mallat, 2013), we show that our construction achieves invariance to rotations while maintaining stability to deformations on the translation group.

Note also that other techniques combining deep neural networks and kernels have been introduced earlier. Multilayer kernel machines were for instance introduced by Cho and Saul (2009); Schölkopf et al. (1998). Shallow kernels for images modeling local regions were also proposed by Schölkopf (1997), and a multilayer construction was proposed by Bo et al. (2011). More recently, different models based on kernels have been introduced by Anselmi et al. (2015); Daniely et al. (2016); Montavon et al. (2011) to gain some theoretical insight about classical multilayer neural networks, while kernels are used by Zhang et al. (2017b) to define convex models for two-layer convolutional networks. Theoretical and practical concerns for learning with multilayer kernels have been studied in Daniely et al. (2017, 2016); Steinwart et al. (2016); Zhang et al. (2016) in addition to CKNs. In particular, Daniely et al. (2017, 2016) study certain classes of dot-product kernels with random feature approximations, Steinwart et al. (2016) consider hierarchical Gaussian kernels with learned weights, and Zhang et al. (2016) study a convex formulation for learning a certain class of fully connected neural networks using a hierarchical kernel. In contrast to these works, our focus is on the kernel *representation* induced by the specific hierarchical kernel defined in CKNs and the geometry of the RKHS. Our characterization of CNNs and activation functions contained in the RKHS is similar to the work of Zhang et al. (2016, 2017b), but differs in several ways: we consider general *homogeneous* dot-product kernels, which yield desirable properties of kernel mappings for stability; we construct generic multi-layer CNNs with pooling in the RKHS, while Zhang et al. (2016) only considers fully-connected networks and Zhang et al. (2017b) is limited to two-layer convolutional networks with no pooling; we quantify the RKHS norm of a CNN depending on its parameters, in particular matrix norms, as a way to control stability and generalization, while Zhang et al. (2016, 2017b) consider models with constrained parameters, and focus on convex learning procedures.

2.1.3 Notation and Basic Mathematical Tools

A positive definite kernel K that operates on a set $\mathcal{X}$ implicitly defines a reproducing kernel Hilbert space $\mathcal{H}$ of functions from $\mathcal{X}$ to $\mathbb{R}$, along with a mapping $\varphi : \mathcal{X} \to \mathcal{H}$. A *predictive model* associates to every point z in $\mathcal{X}$ a label in $\mathbb{R}$; it consists of a linear function f in $\mathcal{H}$ such that $f(z) = \langle f, \varphi(z) \rangle_{\mathcal{H}}$, where $\varphi(z)$ is the *data representation*. Given now two points z, z' in $\mathcal{X}$, Cauchy-Schwarz's inequality allows us to control the

variation of the predictive model f according to the geometry induced by the Hilbert norm $\|.\|_{\mathcal{H}}$:

$$|f(z) - f(z')| \leq \|f\|_{\mathcal{H}} \|\varphi(z) - \varphi(z')\|_{\mathcal{H}}. \tag{2.1}$$

This property implies that two points z and z' that are close to each other according to the RKHS norm should lead to similar predictions, when the model f has small norm in $\mathcal{H}$.

Then, we consider notation from signal processing similar to Mallat (2012). We call a signal x a function in $L^2(\mathbb{R}^d, \mathcal{H})$, where the domain $\mathbb{R}^d$ represents spatial coordinates, and $\mathcal{H}$ is a Hilbert space, when $\|x\|_{L^2}^2 := \int_{\mathbb{R}^d} \|x(u)\|_{\mathcal{H}}^2 du < \infty$, where du is the Lebesgue measure on $\mathbb{R}^d$. Given a linear operator $T : L^2(\mathbb{R}^d, \mathcal{H}) \to L^2(\mathbb{R}^d, \mathcal{H}')$, the operator norm is defined as $\|T\|_{L^2(\mathbb{R}^d,\mathcal{H})\to L^2(\mathbb{R}^d,\mathcal{H}')} := \sup_{\|x\|_{L^2(\mathbb{R}^d,\mathcal{H})}\leq 1} \|Tx\|_{L^2(\mathbb{R}^d,\mathcal{H}')}$. For the sake of clarity, we drop norm subscripts, from now on, using the notation $\|\cdot\|$ for Hilbert space norms, L^2 norms, and $L^2 \to L^2$ operator norms, while $|\cdot|$ denotes the Euclidean norm on $\mathbb{R}^d$. We use cursive capital letters (*e.g.*, $\mathcal{H}, \mathcal{P}$) to denote Hilbert spaces, and non-cursive ones for operators (*e.g.*, P, M, A). Some useful mathematical tools are also presented in Appendix 2.A.

2.1.4 Organization of the Chapter

The rest of the chapter is structured as follows:

- In Section 2.2, we introduce a multilayer convolutional kernel representation for continuous signals, based on a hierarchy of patch extraction, kernel mapping, and pooling operators. We present useful properties of this representation such as signal preservation, as well as ways to make it practical through discretization and kernel approximations in the context of CKNs.

- In Section 2.3, we present our main results regarding stability and invariance, namely that the kernel representation introduced in Section 2.2 is near translation-invariant and stable to the action of diffeomorphisms. We then show in Section 2.3.3 that the same stability results apply in the presence of kernel approximations such as those of CKNs (Mairal, 2016), and describe a generic way to modify the multilayer construction in order to guarantee invariance to the action of any locally compact group of transformations in Section 2.3.5.

- In Section 2.4, we study the functional spaces induced by our representation, showing that simple neural-network like functions with certain smooth activations are contained in the RKHS at intermediate layers, and that the RKHS of the full kernel induced by our representation contains a class of generic CNNs with smooth and homogeneous activations. We then present upper bounds on the RKHS norm of such CNNs, which serves as a measure of complexity, controlling both generalization and stability. Section 2.4.3 studies the stability for CNNs with generic activations such as rectified linear units, and discusses the link with generalization.

- Finally, we discuss in Section 2.5 how the obtained stability results apply to the practical setting of learning prediction functions. In particular, we explain why the regularization used in CKNs provides a natural way to control stability, while a similar control is harder to achieve with generic CNNs.

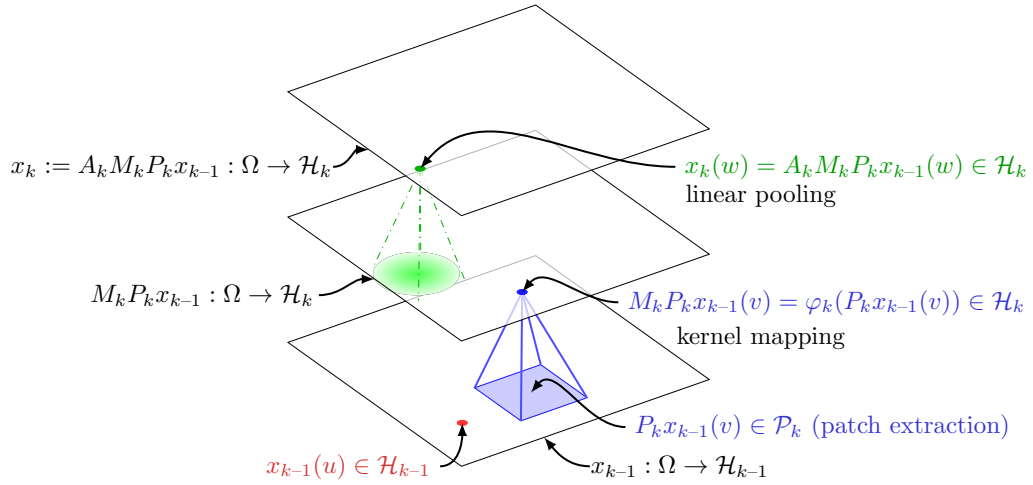


Figure 2.1: Construction of the k-th signal representation from the k–1-th one. Note that while the domain Ω is depicted as a box in $\mathbb{R}^2$ here, our construction is supported on $\Omega = \mathbb{R}^d$.

2.2 Construction of the Multilayer Convolutional Kernel

We now present the multilayer convolutional kernel, which operates on signals with d spatial dimensions. The construction follows closely that of convolutional kernel networks but is generalized to input signals defined on the continuous domain $\mathbb{R}^d$. Dealing with continuous signals is indeed useful to characterize the stability properties of signal representations to small deformations, as done by Mallat (2012) in the context of the scattering transform. The issue of discretization on a discrete grid is addressed in Section 2.2.1.

In what follows, we consider signals x_0 that live in $L^2(\mathbb{R}^d, \mathcal{H}_0)$, where typically $\mathcal{H}_0 = \mathbb{R}^{p_0}$ (*e.g.*, with $p_0 = 3$ and $d = 2$, the vector $x_0(u)$ in $\mathbb{R}^3$ may represent the RGB pixel value at location u in $\mathbb{R}^2$). Then, we build a sequence of reproducing kernel Hilbert spaces $\mathcal{H}_1, \mathcal{H}_2, \ldots$, and transform x_0 into a sequence of "feature maps", respectively denoted by x_1 in $L^2(\mathbb{R}^d, \mathcal{H}_1)$, x_2 in $L^2(\mathbb{R}^d, \mathcal{H}_2)$, *etc...* As depicted in Figure 2.1, a new map x_k is built from the previous one x_{k-1} by applying successively three operators that perform patch extraction (P_k), kernel mapping (M_k) to a new RKHS $\mathcal{H}_k$, and linear pooling (A_k), respectively. When going up in the hierarchy, the points $x_k(u)$ carry information from larger signal neighborhoods centered at u in $\mathbb{R}^d$ with more invariance, as we formally show in Section 2.3.

Patch extraction operator. Given the layer x_{k-1}, we consider a patch shape S_k, defined as a compact centered subset of $\mathbb{R}^d$, *e.g.*, a box, and we define the Hilbert space $\mathcal{P}_k := L^2(S_k, \mathcal{H}_{k-1})$ equipped with the norm $\|z\|^2 = \int_{S_k} \|z(u)\|^2 d\nu_k(u)$, where $d\nu_k$ is the normalized uniform measure on S_k for every z in $\mathcal{P}_k$. Specifically, we define the (linear) patch extraction operator $P_k : L^2(\mathbb{R}^d, \mathcal{H}_{k-1}) \to L^2(\mathbb{R}^d, \mathcal{P}_k)$ such that for all u in $\mathbb{R}^d$,

$$P_k x_{k-1}(u) = (v \mapsto x_{k-1}(u+v))_{v \in S_k} \in \mathcal{P}_k.$$

Note that by equipping $\mathcal{P}_k$ with a normalized measure, it is easy to show that the operator P_k preserves the norm—that is, $\|P_k x_{k-1}\| = \|x_{k-1}\|$ and hence $P_k x_{k-1}$ is in $L^2(\mathbb{R}^d, \mathcal{P}_k)$.

Kernel mapping operator. Then, we map each patch of x_{k-1} to a RKHS $\mathcal{H}_k$ thanks to the kernel mapping $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$ associated to a positive definite kernel K_k that operates on patches. It allows us to define the pointwise operator M_k such that for all u in $\mathbb{R}^d$,

$$M_k P_k x_{k-1}(u) := \varphi_k(P_k x_{k-1}(u)) \in \mathcal{H}_k.$$

In this paper, we consider homogeneous dot-product kernels K_k operating on $\mathcal{P}_k$, defined in terms of a function $\kappa_k : [-1, 1] \to \mathbb{R}$ that satisfies the following constraints:

$$\kappa_k(u) = \sum_{j=0}^{+\infty} b_j u^j \quad \text{s.t.} \quad \forall j, b_j \geq 0, \quad \kappa_k(1) = 1, \quad \kappa_k'(1) = 1, \tag{A1}$$

assuming convergence of the series $\sum_j b_j$ and $\sum_j j b_j$. Then, we define the kernel K_k by

$$K_k(z, z') = \|z\|\|z'\|\kappa_k\left(\frac{\langle z, z'\rangle}{\|z\|\|z'\|}\right), \tag{2.2}$$

if $z, z' \in \mathcal{P}_k \backslash \{0\}$, and $K_k(z, z') = 0$ if $z = 0$ or $z' = 0$. The kernel is positive definite since it admits a Maclaurin expansion with only non-negative coefficients (Schoenberg, 1942; Schölkopf and Smola, 2001). The condition $\kappa_k(1) = 1$ ensures that the RKHS mapping preserves the norm—that is, $\|\varphi_k(z)\| = K_k(z, z)^{1/2} = \|z\|$, and thus $\|M_k P_k x_{k-1}(u)\| = \|P_k x_{k-1}(u)\|$ for all u in $\mathbb{R}^d$; as a consequence, $M_k P_k x_{k-1}$ is always in $L^2(\mathbb{R}^d, \mathcal{H}_k)$. The technical condition $\kappa_k'(1) = 1$, where κ_k' is the first derivative of κ_k, ensures that the kernel mapping φ_k is non-expansive, according to Lemma 2.1 below.

Lemma 2.1 (Non-expansiveness of the kernel mappings)**.** *Consider a positive-definite kernel of the form* (2.2) *satisfying* (A1) *with RKHS mapping $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$. Then, φ_k is non-expansive—that is, for all z, z' in $\mathcal{P}_k$,*

$$\|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\|.$$

Moreover, we remark that the kernel K_k is lower-bounded by the linear one

$$K_k(z, z') \geq \langle z, z'\rangle. \tag{2.3}$$

From the proof of the lemma, given in Appendix 2.B, one may notice that the assumption $\kappa_k'(1) = 1$ is not critical and may be safely replaced by $\kappa_k'(1) \leq 1$. Then, the non-expansiveness property would be preserved. Yet, we have chosen a stronger constraint since it yields a few simplifications in the stability analysis, where we use the relation (2.3) that requires $\kappa_k'(1) = 1$. More generally, the kernel mapping is Lipschitz continuous with constant $\rho_k = \max(1, \sqrt{\kappa_k'(1)})$. Our stability results hold in a setting with $\rho_k > 1$, but with constants $\prod_k \rho_k$ that may grow exponentially with the number of layers.

Examples of functions κ_k that satisfy the properties (A1) are now given below:

exponential	$\kappa_{\exp}(\langle z, z' \rangle) = e^{\langle z, z' \rangle - 1}$
inverse polynomial	$\kappa_{\text{inv-poly}}(\langle z, z' \rangle) = \frac{1}{2 - \langle z, z' \rangle}$
polynomial, degree p	$\kappa_{\text{poly}}(\langle z, z' \rangle) = \frac{1}{(c+1)^p}(c + \langle z, z' \rangle)^p \quad \text{with} \quad c = p - 1$
arc-cosine, degree 1	$\kappa_{\text{acos}}(\langle z, z' \rangle) = \frac{1}{\pi}\left(\sin(\theta) + (\pi - \theta)\cos(\theta)\right)$ with $\theta = \arccos(\langle z, z' \rangle)$
Vovk's, degree 3	$\kappa_{\text{vovk}}(\langle z, z' \rangle) = \frac{1}{3}\left(\frac{1 - \langle z, z' \rangle^3}{1 - \langle z, z' \rangle}\right) = \frac{1}{3}\left(1 + \langle z, z' \rangle + \langle z, z' \rangle^2\right)$

We note that the inverse polynomial kernel was used by Zhang et al. (2016, 2017b) to build convex models of fully connected networks and two-layer convolutional neural networks, while the arc-cosine kernel appears in early deep kernel machines (Cho and Saul, 2009). Note that the homogeneous exponential kernel reduces to the Gaussian kernel for unit-norm vectors. Indeed, for all z, z' such that $\|z\| = \|z'\| = 1$, we have

$$\kappa_{\exp}(\langle z, z' \rangle) = e^{\langle z, z' \rangle - 1} = e^{-\frac{1}{2}\|z - z'\|^2},$$

and thus, we may refer to kernel (2.2) with the function $\kappa_{\exp}$ as the homogeneous Gaussian kernel. The kernel $\kappa(\langle z, z' \rangle) = e^{\alpha(\langle z, z' \rangle - 1)} = e^{-\frac{\alpha}{2}\|z - z'\|^2}$ with $\alpha \neq 1$ may also be used here, but we choose $\alpha = 1$ for simplicity since $\kappa'(1) = \alpha$ (see discussion above).

Pooling operator. The last step to build the layer x_k consists of pooling neighboring values to achieve local shift-invariance. We apply a linear convolution operator A_k with a Gaussian filter of scale σ_k, $h_{\sigma_k}(u) := \sigma_k^{-d} h(u/\sigma_k)$, where $h(u) = (2\pi)^{-d/2} \exp(-|u|^2/2)$. Then, for all u in $\mathbb{R}^d$,

$$x_k(u) = A_k M_k P_k x_{k-1}(u) = \int_{\mathbb{R}^d} h_{\sigma_k}(u - v) M_k P_k x_{k-1}(v) dv \in \mathcal{H}_k, \tag{2.4}$$

where the integral is a Bochner integral (see, Diestel and Uhl, 1977; Muandet et al., 2017). By applying Schur's test to the integral operator A_k (see Appendix 2.A), we obtain that the operator norm $\|A_k\|$ is less than 1. Thus, x_k is in $L^2(\mathbb{R}^d, \mathcal{H}_k)$, with $\|x_k\| \leq \|M_k P_k x_{k-1}\|$. Note that a similar pooling operator is used in the scattering transform (Mallat, 2012).

Multilayer construction and prediction layer. Finally, we obtain a multilayer representation by composing multiple times the previous operators. In order to increase invariance with each layer and to increase the size of the receptive fields (that is, the neighborhood of the original signal considered in a given patch), the size of the patch S_k and pooling scale σ_k typically grow exponentially with k, with σ_k and the patch size $\sup_{c \in S_k} |c|$ of the same order. With n layers, the maps x_n may then be written

$$x_n := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x_0 \ \in \ L^2(\mathbb{R}^d, \mathcal{H}_n). \tag{2.5}$$

It remains to define a kernel from this representation, that will play the same role as the "fully connected" layer of classical convolutional neural networks. For that purpose, we simply consider the following linear kernel defined for all x_0, x_0' in $L^2(\mathbb{R}^d, \mathcal{H}_0)$

by using the corresponding feature maps x_n, x'_n in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ given by our multilayer construction (2.5):

$$\mathcal{K}_n(x_0, x'_0) = \langle x_n, x'_n \rangle = \int_{u \in \mathbb{R}^d} \langle x_n(u), x'_n(u) \rangle du. \tag{2.6}$$

Then, the RKHS $\mathcal{H}_{\mathcal{K}_n}$ of $\mathcal{K}_n$ contains all functions of the form $f(x_0) = \langle w, x_n \rangle$ with w in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ (see Appendix 2.A).

We note that one may also consider nonlinear kernels, such as a Gaussian kernel:

$$\mathcal{K}_n(x_0, x'_0) = e^{-\frac{\alpha}{2}\|x_n - x'_n\|^2}. \tag{2.7}$$

Such kernels are then associated to a RKHS denoted by $\mathcal{H}_{n+1}$, along with a kernel mapping $\varphi_{n+1} : L^2(\mathbb{R}^d, \mathcal{H}_n) \to \mathcal{H}_{n+1}$ which we call *prediction layer*, so that the final representation is given by $\varphi_{n+1}(x_n)$ in $\mathcal{H}_{n+1}$. We note that φ_{n+1} is non-expansive for the Gaussian kernel when $\alpha \leq 1$ (see Section 2.B.1), and is simply an isometric linear mapping for the linear kernel. Then, we have the relation $\mathcal{K}_n(x_0, x'_0) := \langle \varphi_{n+1}(x_n), \varphi_{n+1}(x'_n) \rangle$, and in particular, the RKHS $\mathcal{H}_{\mathcal{K}_n}$ of $\mathcal{K}_n$ contains all functions of the form $f(x_0) = \langle w, \varphi_{n+1}(x_n) \rangle$ with w in $\mathcal{H}_{n+1}$, see Appendix 2.A.

2.2.1 Signal Preservation and Discretization

In this section, we show that the multilayer kernel representation preserves all information about the signal at each layer, and besides, each feature map x_k can be sampled on a discrete set with no loss of information. This suggests a natural approach for discretization which will be discussed after the following lemma, whose proof is given in Appendix 2.C.

Lemma 2.2 (Signal recovery from sampling)**.** *Assume that $\mathcal{H}_k$ contains all linear functions $z \mapsto \langle g, z \rangle$ with g in $\mathcal{P}_k$ (this is true for all kernels K_k described in the previous section, according to Corollary 2.12 in Section 2.4.1 later); then, the signal x_{k-1} can be recovered from a sampling of x_k at discrete locations in a set Ω as soon as $\Omega + S_k = \mathbb{R}^d$ (*i.e.*, the union of patches centered at these points covers $\mathbb{R}^d$). It follows that x_k can be reconstructed from such a sampling.*

The previous construction defines a kernel representation for general signals lying in $L^2(\mathbb{R}^d, \mathcal{H}_0)$, which is an abstract object defined for theoretical purposes. In practice, signals are discrete, and it is thus important to discuss the problem of discretization. For clarity, we limit the presentation to 1-dimensional signals ($d = 1$), but the arguments can easily be extended to higher dimensions d when using box-shaped patches. Notation from the previous section is preserved, but we add a bar on top of all discrete analogues of their continuous counterparts. *e.g.*, $\bar{x}_k$ is a discrete feature map in $\ell^2(\mathbb{Z}, \bar{\mathcal{H}}_k)$ for some RKHS $\bar{\mathcal{H}}_k$.

Input signals x_0 and $\bar{x}_0$. Discrete signals acquired by a physical device may be seen as local integrators of signals defined on a continuous domain (*e.g.*, sensors from digital cameras integrate the pointwise distribution of photons in a spatial and temporal window). Then, consider a signal x_0 in $L^2(\mathbb{R}^d, \mathcal{H}_0)$ and s_0 a sampling interval. By defining $\bar{x}_0$ in $\ell_2(\mathbb{Z}, \mathcal{H}_0)$ such that $\bar{x}_0[n] = x_0(ns_0)$ for all n in $\mathbb{Z}$, it is thus natural

to assume that $x_0 = A_0 x$, where A_0 is a pooling operator (local integrator) applied to an original continuous signal x. The role of A_0 is to prevent aliasing and reduce high frequencies; typically, the scale σ_0 of A_0 should be of the same magnitude as s_0, which we choose to be $s_0 = 1$ without loss of generality. This natural assumption is kept later for the stability analysis.

Multilayer construction. We now want to build discrete feature maps $\bar{x}_k$ in $\ell^2(\mathbb{Z}, \bar{\mathcal{H}}_k)$ at each layer k involving subsampling with a factor s_k with respect to $\bar{x}_{k-1}$. We now define the discrete analogues of the operators P_k (patch extraction), M_k (kernel mapping), and A_k (pooling) as follows: for $n \in \mathbb{Z}$,

$$\begin{aligned}
\bar{P}_k \bar{x}_{k-1}[n] &:= \frac{1}{\sqrt{e_k}}(\bar{x}_{k-1}[n], \bar{x}_{k-1}[n+1], \ldots, \bar{x}_{k-1}[n+e_k-1]) \in \bar{\mathcal{P}}_k := \bar{\mathcal{H}}_{k-1}^{e_k} \\
\bar{M}_k \bar{P}_k \bar{x}_{k-1}[n] &:= \bar{\varphi}_k(\bar{P}_k \bar{x}_{k-1}[n]) \in \bar{\mathcal{H}}_k \\
\bar{x}_k[n] = \bar{A}_k \bar{M}_k \bar{P}_k \bar{x}_{k-1}[n] &:= \frac{1}{\sqrt{s_k}} \sum_{m \in \mathbb{Z}} \bar{h}_k[n s_k - m] \bar{M}_k \bar{P}_k \bar{x}_{k-1}[m] = (\bar{h}_k * \bar{M}_k \bar{P}_k \bar{x}_{k-1})[n s_k] \in \bar{\mathcal{H}}_k,
\end{aligned}$$

where (i) $\bar{P}_k$ extracts a patch of size e_k starting at position n in $\bar{x}_{k-1}[n]$, which lives in the Hilbert space $\bar{\mathcal{P}}_k$ defined as the direct sum of e_k times $\bar{\mathcal{H}}_{k-1}$; (ii) $\bar{M}_k$ is a kernel mapping identical to the continuous case, which preserves the norm, like M_k; (iii) $\bar{A}_k$ performs a convolution with a Gaussian filter and a subsampling operation with factor s_k. The next lemma shows that under mild assumptions, this construction preserves signal information.

Lemma 2.3 (Signal recovery with subsampling)**.** *Assume that $\bar{\mathcal{H}}_k$ contains the linear functions $z \mapsto \langle w, z \rangle$ for all w in $\bar{\mathcal{P}}_k$ and that $e_k \geq s_k$. Then, $\bar{x}_{k-1}$ can be recovered from $\bar{x}_k$.*

The proof is given in Appendix 2.C. The result relies on recovering patches using linear "measurement" functions and deconvolution of the pooling operation. While such a deconvolution operation can be unstable, it may be possible to obtain more stable recovery mechanisms by also considering non-linear measurements, a question which we leave open.

Links between the parameters of the discrete and continuous models. Due to subsampling, the patch size in the continuous and discrete models are related by a multiplicative factor. Specifically, a patch of size e_k with discretization corresponds to a patch S_k of diameter $e_k s_{k-1} s_{k-2} \ldots s_1$ in the continuous case. The same holds true for the scale parameter σ_k of the Gaussian pooling.

2.2.2 Practical Implementation via Convolutional Kernel Networks

Besides discretization, convolutional kernel networks add two modifications to implement in practice the image representation we have described. First, it uses feature maps with finite spatial support, which introduces border effects that we do not study (like Mallat, 2012), but which are negligible when dealing with large realistic images. Second, CKNs use finite-dimensional approximations of the kernel feature map. Typically, each

RKHS's mapping is approximated by performing a projection onto a subspace of finite dimension, which is a classical approach to make kernel methods work at large scale (Fine and Scheinberg, 2001; Smola and Schölkopf, 2000; Williams and Seeger, 2001). If we consider the kernel mapping $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$ at layer k, the orthogonal projection onto the finite-dimensional subspace $\mathcal{F}_k = \text{span}(\varphi_k(z_1), \dots, \varphi_k(z_{p_k})) \subseteq \mathcal{H}_k$, where the z_i's are p_k anchor points in $\mathcal{P}_k$, is given by the linear operator $\Pi_k : \mathcal{H}_k \to \mathcal{F}_k$ defined for f in $\mathcal{H}_k$ by

$$\Pi_k f := \sum_{1 \leq i,j \leq p_k} (K_{ZZ}^{-1})_{ij} \langle \varphi_k(z_i), f \rangle \varphi_k(z_j), \tag{2.8}$$

where K_{ZZ}^{-1} is the inverse (or pseudo-inverse) of the $p_k \times p_k$ kernel matrix $[K_k(z_i, z_j)]_{ij}$. As an orthogonal projection operator, Π_k is non-expansive, *i.e.*, $\|\Pi_k\| \leq 1$. We can then define the new approximate version $\tilde{M}_k$ of the kernel mapping operator M_k by

$$\tilde{M}_k P_k x_{k-1}(u) := \Pi_k \varphi_k(P_k x_{k-1}(u)) \in \mathcal{F}_k. \tag{2.9}$$

Note that all points in the feature map $\tilde{M}_k P_k x_{k-1}$ lie in the p_k-dimensional space $\mathcal{F}_k \subseteq \mathcal{H}_k$, which allows us to represent each point $\tilde{M}_k P_k x_{k-1}(u)$ by the finite dimensional vector

$$\psi_k(P_k x_{k-1}(u)) := K_{ZZ}^{-1/2} K_Z(P_k x_{k-1}(u)) \in \mathbb{R}^{p_k}, \tag{2.10}$$

with $K_Z(z) := (K_k(z_1, z), \dots, K_k(z_{p_k}, z))^\top$; this finite-dimensional representation preserves the Hilbertian inner product and norm[1] in $\mathcal{F}_k$ so that we have

$$\|\psi_k(P_k x_{k-1}(u))\|_2^2 = \|\tilde{M}_k P_k x_{k-1}(u)\|_{\mathcal{H}_k}^2.$$

Such a finite-dimensional mapping is compatible with the multilayer construction, which builds $\mathcal{H}_k$ by manipulating points from $\mathcal{H}_{k-1}$. Here, the approximation provides points in $\mathcal{F}_k \subseteq \mathcal{H}_k$, which remain in $\mathcal{F}_k$ after pooling since $\mathcal{F}_k$ is a linear subspace. Eventually, the sequence of RKHSs $\{\mathcal{H}_k\}_{k \geq 0}$ is not affected by the finite-dimensional approximation. Besides, the stability results we will present next are preserved thanks to the non-expansiveness of the projection. In contrast, other kernel approximations such as random Fourier features (Rahimi and Recht, 2007) do not provide points in the RKHS (see Bach, 2017b), and their effect on the functional space derived from the multilayer construction is unclear.

It is then possible to derive theoretical results for the CKN model, which appears as a natural implementation of the kernel constructed previously; yet, we will also show in Section 2.4 that the results apply more broadly to CNNs that are contained in the functional space associated to the kernel. However, the stability of these CNNs depends on their RKHS norm, which is hard to control. In contrast, for CKNs, stability is typically controlled by the norm of the final prediction layer.

2.3 Stability to Deformations and Group Invariance

In this section, we study the translation invariance and the stability under the action of diffeomorphisms of the kernel representation described in Section 2.2 for continuous signals. In addition to translation invariance, it is desirable to have a representation

[1] We have $\langle \psi_k(z), \psi_k(z') \rangle_2 = \langle \Pi_k \varphi_k(z), \Pi_k \varphi_k(z') \rangle_{\mathcal{H}_k}$. See Mairal (2016) for details.

that is stable to small local deformations. We describe such deformations using a C^1-diffeomorphism $\tau : \mathbb{R}^d \to \mathbb{R}^d$, and let L_τ denote the linear operator defined by $L_\tau x(u) = x(u - \tau(u))$. We use a similar characterization of stability to the one introduced by Mallat (2012): the representation $\Phi(\cdot)$ is *stable* under the action of diffeomorphisms if there exist two non-negative constants C_1 and C_2 such that

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1 \|\nabla\tau\|_\infty + C_2 \|\tau\|_\infty)\|x\|, \tag{2.11}$$

where $\nabla\tau$ is the Jacobian of τ, $\|\nabla\tau\|_\infty := \sup_{u\in\mathbb{R}^d} \|\nabla\tau(u)\|$, and $\|\tau\|_\infty := \sup_{u\in\mathbb{R}^d} |\tau(u)|$. The quantity $\|\nabla\tau(u)\|$ measures the size of the deformation at a location u, and like Mallat (2012), we assume the regularity condition $\|\nabla\tau\|_\infty \leq 1/2$, which implies that the deformation is invertible (Allassonnière et al., 2007; Trouvé and Younes, 2005) and helps us avoid degenerate situations. In order to have a near-translation-invariant representation, we want C_2 to be small (a translation is a diffeomorphism with $\nabla\tau = 0$), and indeed we will show that C_2 is proportional to $1/\sigma_n$, where σ_n is the scale of the last pooling layer, which typically increases exponentially with the number of layers n. When $\nabla\tau$ is non-zero, the diffeomorphism deviates from a translation, producing local deformations controlled by $\nabla\tau$.

Additional assumptions. In order to study the stability of the representation (2.5), we assume that the input signal x_0 may be written as $x_0 = A_0 x$, where A_0 is an initial pooling operator at scale σ_0, which allows us to control the high frequencies of the signal in the first layer. As discussed previously in Section 2.2.1, this assumption is natural and compatible with any physical acquisition device. Note that σ_0 can be taken arbitrarily small, so that this assumption does not limit the generality of our results. Then, we are interested in understanding the stability of the representation

$$\Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x.$$

We do not consider a prediction layer φ_{n+1} here for simplicity, but note that if we add one on top of Φ_n, based on a linear of Gaussian kernel, then the stability of the full representation $\varphi_{n+1} \circ \Phi_n$ immediately follows from that of Φ_n thanks to the non-expansiveness of φ_{n+1} (see Section 2.2). Then, we make an assumption that relates the scale of the pooling operator at layer $k-1$ with the diameter of the patch S_k: we assume indeed that there exists $\kappa > 0$ such that for all $k \geq 1$,

$$\sup_{c\in S_k} |c| \leq \kappa \sigma_{k-1}. \tag{A2}$$

The scales σ_k are typically exponentially increasing with the layers k, and characterize the "resolution" of each feature map. This assumption corresponds to considering patch sizes that are adapted to these intermediate resolutions. Moreover, the stability bounds we obtain hereafter increase with κ, which leads us to believe that small patch sizes lead to more stable representations, something which matches well the trend of using small, 3x3 convolution filters at each scale in modern deep architectures (*e.g.*, Simonyan and Zisserman, 2014).

Finally, before presenting our stability results, we recall a few properties of the operators involved in the representation Φ_n, which are heavily used in the analysis.

1. **Patch extraction operator**: P_k is linear and preserves the norm;
2. **Kernel mapping operator**: M_k preserves the norm and is non-expansive;
3. **Pooling operator**: A_k is linear and non-expansive $\|A_k\| \leq 1$;

The rest of this section is organized into three parts. We present the main stability results in Section 2.3.1, explain their compatibility with kernel approximations in Section 2.3.3, and provide numerical experiment for demonstrating the stability of the kernel representation in Section 2.3.4. Finally, we introduce mechanisms to achieve invariance to any group of transformations in Section 2.3.5.

2.3.1 Stability Results and Translation Invariance

Here, we show that our kernel representation Φ_n satisfies the stability property (2.11), with a constant C_2 inversely proportional to σ_n, thereby achieving near-invariance to translations. The results are then extended to more general transformation groups in Section 2.3.5.

General bound for stability. The following result gives an upper bound on the quantity of interest, $\|\Phi_n(L_\tau x) - \Phi_n(x)\|$, in terms of the norm of various linear operators which control how τ affects each layer. An important object of study is the commutator of linear operators A and B, which is denoted by $[A, B] = AB - BA$.

Proposition 2.4 (Bound with operator norms)**.** *For any x in $L^2(\mathbb{R}^d, \mathcal{H}_0)$, we have*

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(\sum_{k=1}^{n} \|[P_k A_{k-1}, L_\tau]\| + \|[A_n, L_\tau]\| + \|L_\tau A_n - A_n\| \right) \|x\|. \quad (2.12)$$

For translations $L_\tau x(u) = L_c x(u) = x(u - c)$, it is easy to see that patch extraction and pooling operators commute with L_c (this is also known as *covariance* or *equivariance* to translations), so that we are left with the term $\|L_c A_n - A_n\|$, which should control translation invariance. For general diffeomorphisms τ, we no longer have exact covariance, but we show below that commutators are stable to τ, in the sense that $\|[P_k A_{k-1}, L_\tau]\|$ is controlled by $\|\nabla \tau\|_\infty$, while $\|L_\tau A_n - A_n\|$ is controlled by $\|\tau\|_\infty$ and decays with the pooling size σ_n.

Bound on $\|[P_k A_{k-1}, L_\tau]\|$. We note that $P_k z$ can be identified with $(L_c z)_{c \in S_k}$ isometrically for all z in $L^2(\mathbb{R}^d, \mathcal{H}_{k-1})$, since $\|P_k z\|^2 = \int_{S_k} \|L_c z\|^2 d\nu_k(c)$ by Fubini's theorem. Then,

$$\begin{aligned} \|P_k A_{k-1} L_\tau z - L_\tau P_k A_{k-1} z\|^2 &= \int_{S_k} \|L_c A_{k-1} L_\tau z - L_\tau L_c A_{k-1} z\|^2 d\nu_k(c) \\ &\leq \sup_{c \in S_k} \|L_c A_{k-1} L_\tau z - L_\tau L_c A_{k-1} z\|^2, \end{aligned}$$

so that $\|[P_k A_{k-1}, L_\tau]\| \leq \sup_{c \in S_k} \|[L_c A_{k-1}, L_\tau]\|$. The following result lets us bound the commutator $\|[L_c A_{k-1}, L_\tau]\|$ when $|c| \leq \kappa \sigma_{k-1}$, which is satisfied under assumption (A2).

Lemma 2.5 (Stability of shifted pooling)**.** *Consider A_σ the pooling operator with kernel $h_\sigma(u) = \sigma^{-d}h(u/\sigma)$. If $\|\nabla\tau\|_\infty \leq 1/2$, there exists a constant C_1 such that for any σ and $|c| \leq \kappa\sigma$, we have*

$$\|[L_c A_\sigma, L_\tau]\| \leq C_1 \|\nabla\tau\|_\infty,$$

where C_1 depends only on h and κ.

A similar result can be found in Lemma E.1 of Mallat (2012) for commutators of the form $[A_\sigma, L_\tau]$, but we extend it to handle integral operators $L_c A_\sigma$ with a shifted kernel. The proof (given in Appendix 2.C.4) follows closely Mallat (2012) and relies on the fact that $[L_c A_\sigma, L_\tau]$ is an integral operator in order to bound its norm via Schur's test. Note that κ can be made larger, at the cost of an increase of the constant C_1 of the order κ^{d+1}.

Bound on $\|L_\tau A_n - A_n\|$. We bound the operator norm $\|L_\tau A_n - A_n\|$ in terms of $\|\tau\|_\infty$ using the following result due to Mallat (2012, Lemma 2.11), with $\sigma = \sigma_n$:

Lemma 2.6 (Translation invariance)**.** *If $\|\nabla\tau\|_\infty \leq 1/2$, we have*

$$\|L_\tau A_\sigma - A_\sigma\| \leq \frac{C_2}{\sigma}\|\tau\|_\infty,$$

with $C_2 = 2^d \cdot \|\nabla h\|_1$.

Combining Proposition 2.4 with Lemmas 2.5 and 2.6, we obtain the following result:

Theorem 2.7 (Stability bound)**.** *Assume* (A2)*. If $\|\nabla\tau\|_\infty \leq 1/2$, we have*

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(C_1\,(1+n)\,\|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n}\|\tau\|_\infty\right)\|x\|. \tag{2.13}$$

This result matches the desired notion of stability in Eq. (2.11), with a translation-invariance factor that decays with σ_n. We discuss implications of our bound, and compare it with related work on stability in Section 2.3.2. We also note that our bound yields a worst-case guarantee on stability, in the sense that it holds for any signal x. In particular, making additional assumptions on the signal (*e.g.*, smoothness) may lead to improved stability. The predictions for a specific model may also be more stable than applying (2.1) to our stability bound, for instance if the filters are smooth enough.

Remark 2.8 (Stability for Lipschitz non-linear mappings)**.** *While the previous results require non-expansive non-linear mappings φ_k, it is easy to extend the result to the following more general condition*

$$\|\varphi_k(z) - \varphi_k(z')\| \leq \rho_k\|z - z'\| \quad \text{and} \quad \|\varphi_k(z)\| \leq \rho_k\|z\|.$$

Indeed, the proof of Proposition 2.4 easily extends to this setting, giving an additional factor $\prod_k \rho_k$ in the bound (2.11)*. The stability bound* (2.13) *then becomes*

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(\prod_{k=1}^{n}\rho_k\right)\left(C_1\,(1+n)\,\|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n}\|\tau\|_\infty\right)\|x\|. \tag{2.14}$$

This will be useful for obtaining stability of CNNs with generic activations such as ReLU (see Section 2.4.3), and this also captures the case of kernels with $\kappa_k'(1) > 1$ in Lemma 2.1.

2.3.2 Discussion of the Stability Bound (Theorem 2.7)

In this section, we discuss the implications of our stability bound (2.13), and compare it to related work on the stability of the scattering transform (Mallat, 2012) as well as the work of (Wiatowski and Bölcskei, 2018) on more general convolutional models.

Role of depth. Our bound displays a linear dependence on the number of layers n in the stability constant $C_1(1+n)$. We note that a dependence on a notion of depth (the number of layers n here) also appears in Mallat (2012), with a factor equal to the maximal length of "scattering paths", and with the same condition $\|\nabla\tau\|_\infty \leq 1/2$. Nevertheless, the number of layers is tightly linked to the patch sizes, and we now show how a deeper architecture can be beneficial for stability. Given a desired level of translation-invariance σ_f and a given initial resolution σ_0, the above bound together with the discretization results of Section 2.2.1 suggest that one can obtain a stable representation that preserves signal information by taking small patches at each layer and subsampling with a factor equal to the patch size (assuming a patch size greater than one) until the desired level of invariance is reached: in this case we have $\sigma_f/\sigma_0 \approx \kappa^n$, where κ is of the order of the patch size, so that $n = O(\log(\sigma_f/\sigma_0)/\log(\kappa))$, and hence the stability constant $C_1(1+n)$ grows with κ as $\kappa^{d+1}/\log(\kappa)$, explaining the benefit of small patches, and thus of deeper models.

Norm preservation. While the scattering representation preserves the norm of the input signals when the length of scattering paths goes to infinity, in our setting the norm may decrease with depth due to pooling layers. However, we show in Appendix 2.C.5 that a part of the signal norm is still preserved, particularly for signals with high energy in the low frequencies, as is the case for natural images (*e.g.*, Torralba and Oliva, 2003). This justifies that the bounded quantity in (2.13) is relevant and non-trivial. Nevertheless, we recall that despite a possible loss in norm, our (infinite-dimensional) representation $\Phi(x)$ preserves signal information, as discussed in Section 2.2.1.

Dependence on signal bandwidth. We note that our stability result crucially relies on the assumption $\sigma_0 > 0$, which effectively limits its applicability to signals with frequencies bounded by $\lambda_0 \approx 1/\sigma_0$. While this assumption is realistic in practice for digital signals, our bound degrades as σ_0 approaches 0, since the number of layers n grows as $\log(1/\sigma_0)$, as explained above. This is in contrast to the stability bound of Mallat (2012), which holds uniformly over any such σ_0, thanks to the use of more powerful tools from harmonic analysis such as the Cotlar-Stein lemma, which allows to control stability simultaneously at all frequencies thanks to the structure of the wavelet transform, something which seems more challenging in our case due to the non-linearities separating different scales.

We note that it may be difficult to obtain meaningful stability results for an unbounded frequency support given a fixed architecture, without making assumptions about the filters of a specific model. In particular, if we consider a model with a high frequency Fourier or cosine filter at the first layer, supported on a large enough patch relative to the corresponding wavelength, this will cause instabilities, particularly if the input signal has isolated high frequencies (see, *e.g.*, Bruna and Mallat, 2013). By the

arguments of Section 2.4, such an unstable model g is in the RKHS, and we then have that the final representation $\Phi(\cdot)$ is also unstable, since

$$\begin{aligned}\|\Phi(L_\tau x) - \Phi(x)\| &= \sup_{f \in \mathcal{H}_{\mathcal{K}_n}, \|f\| \leq 1} \langle f, \Phi(L_\tau x) - \Phi(x) \rangle \\ &\geq \frac{1}{\|g\|} \langle g, \Phi(L_\tau x) - \Phi(x) \rangle = \frac{1}{\|g\|} (g(L_\tau x) - g(x)).\end{aligned}$$

Comparison with Wiatowski and Bölcskei (2018). The work of Wiatowski and Bölcskei (2018) also studies deformation stability for generic convolutional network models, however their "deformation sensitivity" result only shows that the representation is as sensitive to deformations as the original signal, something which is also applicable here thanks to the non-expansiveness of our representation. Moreover, their bound does not show the dependence on deformation size (the Jacobian norm), and displays a translation invariance part that degrades linearly with $1/\sigma_0$. In contrast, the translation invariance part of our bound is independent of σ_0, and the overall bound only depends logarithmically on $1/\sigma_0$, by exploiting architectural choices such as pooling layers and patch sizes.

2.3.3 Stability with Kernel Approximations

As in the analysis of the scattering transform of Mallat (2012), we have characterized the stability and shift-invariance of the data representation for continuous signals, in order to give some intuition about the properties of the corresponding discrete representation, which we have described in Section 2.2.1.

Another approximation performed in the CKN model of Mairal (2016) consists of adding projection steps on finite-dimensional subspaces of the RKHS's layers, as discusssed in Section 2.2.2. Interestingly, the stability properties we have obtained previously are compatible with these steps. We may indeed replace the operator M_k with the operator $\tilde{M}_k z(u) = \Pi_k \varphi_k(z(u))$ for any map z in $L^2(\mathbb{R}^d, \mathcal{P}_k)$, instead of $M_k z(u) = \varphi_k(z(u))$; $\Pi_k : \mathcal{H}_k \to \mathcal{F}_k$ is here an orthogonal projection operator onto a linear subspace, given in (2.8). Then, $\tilde{M}_k$ does not necessarily preserve the norm anymore, but $\|\tilde{M}_k z\| \leq \|z\|$, with a loss of information equal to $\|M_k z - \tilde{M}_k z\|$ corresponding to the quality of approximation of the kernel K_k on the points $z(u)$. On the other hand, the non-expansiveness of M_k is satisfied thanks to the non-expansiveness of the projection. In summary, it is possible to show that the conclusions of Theorem 2.7 remain valid when adding the CKN projection steps at each layer, but some signal information is lost in the process.

2.3.4 Empirical Study of Stability

In this section, we provide numerical experiments to demonstrate the stability properties of the kernel representations defined in Section 2.2 on discrete images.

We consider images of handwritten digits from the Infinite MNIST dataset of Loosli et al. (2007), which consists of 28x28 grayscale MNIST digits augmented with small translations and deformations. Translations are chosen at random from one of eight possible directions, while deformations are generated by considering small smooth deformations τ, and approximating $L_\tau x$ using a tangent vector field ∇x containing partial

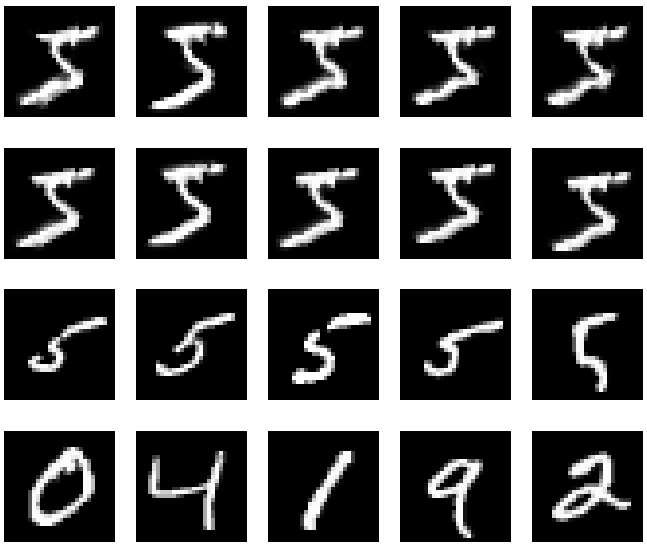


Figure 2.2: MNIST digits with transformations considered in our numerical study of stability. Each row gives examples of images from a set of digits that are compared to a reference image of a "5". From top to bottom: deformations with $\alpha = 3$; translations and deformations with $\alpha = 1$; digits from the training set with the same label "5" as the reference digit; digits from the training set with any label.

derivatives of the signal x along the horizontal and vertical image directions. We introduce a deformation parameter α to control such deformations, which are then given by

$$L_{\alpha\tau}x(u) = x(u - \alpha\tau(u)) \approx x(u) - \alpha\tau(u) \cdot \nabla x(u).$$

Figure 2.2 shows examples of different deformations, with various values of α, with or without translations, generated from a reference image of the digit "5". In addition, one may consider that a given reference image of a handwritten digit can be deformed into different images of the same digit, and perhaps even into a different digit (*e.g.*, a "1" may be deformed into a "7"). Intuitively, the latter transformation corresponds to a "larger" deformation than the former, so that a prediction function that is stable to deformations should be preferable for a classification task. The aim of our experiments is to quantify this stability, and to study how it is affected by architectural choices such as patch sizes and pooling scales.

We consider a full kernel representation, discretized as described in Section 2.2.1. We limit ourselves to 2 layers in order to make the computation of the full kernel tractable. Patch extraction is performed with zero padding in order to preserve the size of the previous feature map. We use a homogeneous dot-product kernel as in Eq. (2.2) with $\kappa(z) = e^{\rho(z-1)}$, $\rho = 1/(0.65)^2$. Note that this choice yields $\kappa'(z) = \rho > 1$, giving an ρ-Lipschitz kernel mapping instead of a non-expansive one as in Lemma 2.1 which considers $\rho = 1$. However, values of ρ larger than one typically lead to better empirical performance for classification (Mairal, 2016), and the stability results of Section 2.3 are still valid with an additional factor ρ^n (with $n = 2$ here) in Eq. (2.13). For a subsampling factor s, we apply a Gaussian filter with scale $\sigma = s/\sqrt{2}$ before downsampling. Our

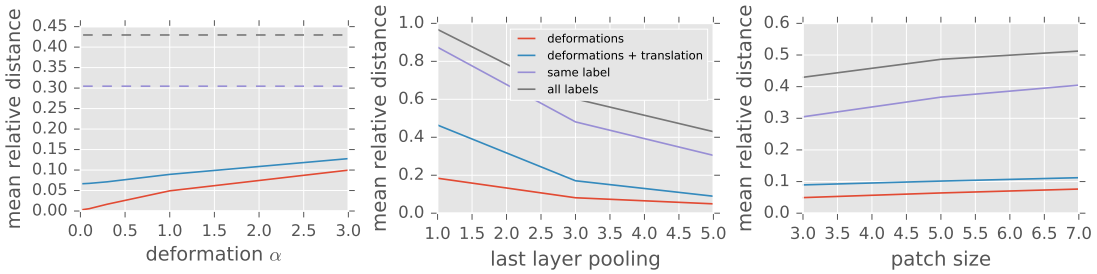


Figure 2.3: Average relative representation distance for various 2-layer models. Lines in the legend corresponds to rows of images in Figure 2.2. In (b-c), deformations are obtained with $\alpha = 1$. We show the impact on relative distance of: (a) the value of α in deformations, in $\{0.01, 0.03, 0.1, 0.3, 1, 3\}$; (b) the subsampling factor of the final pooling layer, in $\{1, 3, 5\}$; (c) the patch size, in $\{3, 5, 7\}$.

C++ implementation for computing the full kernel given two images is available at `https://github.com/albietz/ckn_kernel`.

In Figure 2.3, we show average relative distance in representation space between a reference image and images from various sets of 20 images (either generated transformations, or images appearing in the training set). For a given architecture A and set S of images, the average relative distance to an image x is given by

$$\frac{1}{|S|}\sum_{x'\in S}\frac{\|\Phi_A(x') - \Phi_A(x)\|}{\|\Phi_A(x)\|} = \frac{1}{|S|}\sum_{x'\in S}\frac{\sqrt{K_A(x,x) + K_A(x',x') - 2K_A(x,x')}}{\sqrt{K_A(x,x)}},$$

where Φ_A denotes the kernel representation for architecture A and $K_A(x, x')$ the corresponding kernel. We normalize by $\|\Phi_A(x)\|$ in order to reduce sensitivity to the choice of architecture. We start with a $(3, 2)$-layer followed by a $(3, 5)$-layer, where (p, s) indicates a layer with patch size p and subsampling s. In Figure 2.3b, we vary the subsampling factor of the second layer, and in Figure 2.3c we vary the patch size of both layers.

Each row of Figure 2.2 shows digits and deformed versions. Intuitively, it should be easier to deform an image of a handwritten 5 into a different image of a 5, than into a different digit. Indeed, Figure 2.3 shows that the average relative distance for images with different labels is always larger than for images with the same label, which in turn is larger than for small deformations and translations of the reference image.

Adding translations on top of deformations increases distance in all cases, and Figure 2.3b shows that this gap is smaller when using larger subsampling factors in the last layer. This agrees with the stability bound (2.13), which shows that a larger pooling scale at the last layer increases translation invariance. Figure 2.3a highlights the dependence of the distance on the deformation size α, which is near-linear as in Eq. (2.13) (note that α controls the Jacobian of the deformation). Finally, Figure 2.3c shows that larger patch sizes can make the representations less stable, as discussed in Section 2.3.

2.3.5 Global Invariance to Group Actions

In Section 2.3.1, we have seen how the kernel representation of Section 2.2 creates invariance to translations by commuting with the action of translations at intermediate

layers, and how the last pooling layer on the translation group governs the final level of invariance. It is often useful to encode invariances to different groups of transformations, such as rotations or reflections (see, *e.g.*, Cohen and Welling, 2016; Mallat, 2012; Raj et al., 2017; Sifre and Mallat, 2013). Here, we show how this can be achieved by defining adapted patch extraction and pooling operators that commute with the action of a transformation group G (this is known as group covariance or equivariance). We assume that G is locally compact such that we can define a left-invariant Haar measure μ—that is, a measure on G that satisfies $\mu(gS) = \mu(S)$ for any Borel set $S \subseteq G$ and g in G. We assume the initial signal $x(u)$ is defined on G, and we define subsequent feature maps on the same domain. The action of an element g in G is denoted by L_g, where $L_g x(u) = x(g^{-1}u)$. In order to keep the presentation simple, we ignore some issues related to the general construction in $L^2(G)$ of our signals and operators, which can be made more precise using tools from abstract harmonic analysis (*e.g.*, Folland, 2016).

Extending a signal on G. We note that the original signal is defined on a domain $\mathbb{R}^d$ which may be different from the transformation group G that acts on $\mathbb{R}^d$ (*e.g.*, for 2D images the domain is $\mathbb{R}^2$ but G may also include a rotation angle). The action of g in G on the original signal defined on $\mathbb{R}^d$, denoted $\tilde{x}(\omega)$ yields a transformed signal $L_g\tilde{x}(\omega) = \tilde{x}(g^{-1}\cdot\omega)$, where $\cdot$ denotes group action. This requires an appropriate extension of the signal to G that preserves the meaning of signal transformations. We make the following assumption: every element ω in $\mathbb{R}^d$ can be reached with a transformation u_ω in G from a neutral element ϵ in $\mathbb{R}^d$ (*e.g.*, $\epsilon = 0$), as $\omega = u_\omega \cdot \epsilon$. Note that for 2D images ($d = 2$), this typically requires a group G that is "larger" than translations, such as the roto-translation group, while it is not satisfied, for instance, for rotations only. A similar assumption is made by Kondor and Trivedi (2018). Then, one can extend the original signal $\tilde{x}$ by defining $x(u) := \tilde{x}(u \cdot \epsilon)$. Indeed, we then have

$$L_g x(u_\omega) = x(g^{-1}u_\omega) = \tilde{x}((g^{-1}u_\omega)\cdot\epsilon) = \tilde{x}(g^{-1}\cdot\omega),$$

so that the signal $(x(u_\omega))_{\omega\in\mathbb{R}^d}$ preserves the structure of $\tilde{x}$. We detail this below for the example of roto-translations on 2D images. Then, we are interested in defining a layer—that is, a succession of patch extraction, kernel mapping, and pooling operators—that commutes with L_g, in order to achieve equivariance to G.

Patch extraction. We define patch extraction as follows

$$Px(u) = (x(uv))_{v\in S} \quad \text{for all} \quad u \in G,$$

where $S \subset G$ is a patch shape centered at the identity element. P commutes with L_g since

$$PL_g x(u) = (L_g x(uv))_{v\in S} = (x(g^{-1}uv))_{v\in S} = Px(g^{-1}u) = L_g Px(u).$$

Kernel mapping. The pointwise operator M is defined exactly as in Section 2.2, and thus commutes with L_g.

Pooling. The pooling operator on the group G is defined by

$$Ax(u) = \int_G x(uv)h(v)d\mu(v) = \int_G x(v)h(u^{-1}v)d\mu(v),$$

where h is a pooling filter typically localized around the identity element. The construction is similar to Raj et al. (2017) and it is easy to see from the first expression of $Ax(u)$ that $AL_g x(u) = L_g Ax(u)$, making the pooling operator G-equivariant. One may also pool on a subset of the group by only integrating over the subset in the first expression, an operation which is also G-equivariant.

In our analysis of stability in Section 2.3.1, we saw that inner pooling layers are useful to guarantee stability to local deformations, while global invariance is achieved mainly through the last pooling layer. In some cases, one only needs stability to a subgroup of G, while achieving invariance to the whole group, *e.g.*, in the roto-translation group (Oyallon and Mallat, 2015; Sifre and Mallat, 2013), one might want invariance to a global rotation but stability to local translations. Then, one can perform patch extraction and pooling just on the subgroup to stabilize (*e.g.*, translations) in intermediate layers, while pooling on the entire group at the last layer to achieve the global group invariance.

Example with the roto-translation group. We consider a simple example on 2D images where one wants global invariance to rotations in addition to near-invariance and stability to translations as in Section 2.3.1. For this, we consider the roto-translation group (see, *e.g.*, Sifre and Mallat, 2013), defined as the *semi-direct* product of translations $\mathbb{R}^2$ and rotations $SO(2)$, denoted by $G = \mathbb{R}^2 \rtimes SO(2)$, with the following group operation

$$gg' = (v + R_\theta v', \theta + \theta'),$$

for $g = (v, \theta)$, $g' = (v', \theta')$ in G, where R_θ is a rotation matrix in $SO(2)$. The element $g = (v, \theta)$ in G acts on a location $u \in \mathbb{R}^2$ by combining a rotation and a translation:

$$\begin{aligned} g \cdot u &= v + R_\theta u \\ g^{-1} \cdot u &= (-R_{-\theta} v, -\theta) \cdot u = R_{-\theta}(u - v). \end{aligned}$$

For a given image $\tilde{x}$ in $L^2(\mathbb{R}^2)$, our equivariant construction outlined above requires an extension of the signal to the group G. We consider the Haar measure given by $d\mu((v, \theta)) := dv d\mu_c(\theta)$, where dv is the Lebesgue measure on $\mathbb{R}^2$ and $d\mu_c$ the normalized Haar measure on the unit circle. Note that μ is left-invariant, since the determinant of rotation matrices that appears in the change of variables is 1. We can then define x by $x((u, \eta)) := \tilde{x}(u)$ for any angle η, which is in $L^2(G)$ and preserves the definition of group action on the original signal $\tilde{x}$ since

$$L_g x((u, \eta)) = x(g^{-1}(u, \eta)) = x((g^{-1} \cdot u, \eta - \theta)) = \tilde{x}(g^{-1} \cdot u) = L_g \tilde{x}(u).$$

That is, we can study the action of G on 2D images in $L^2(\mathbb{R}^2)$ by studying the action on the extended signals in $L^2(G)$ defined above.

We can now define patch extraction and pooling operators $P, A : L^2(G) \to L^2(G)$ only on the translation subgroup, by considering a patch shape $S = \{(v, 0)\}_{v \in \tilde{S}} \subset G$ with $\tilde{S} \subset \mathbb{R}^2$ for P, and defining pooling by $Ax(g) = \int_{\mathbb{R}^d} x(g(v, 0))h(v)dv$, where h is a Gaussian pooling filter with scale σ defined on $\mathbb{R}^2$.

The following result, proved in Appendix 2.C, shows analogous results to the stability lemmas of Section 2.3.1 for the operators P and A. For a diffeomorphism τ, we denote by L_τ the action operator given by $L_\tau x((u,\eta)) = x((\tau(u),0)^{-1}(u,\eta)) = x((u-\tau(u),\eta))$.

Lemma 2.9 (Stability with roto-translation patches)**.** *If $\|\nabla\tau\|_\infty \leq 1/2$, and the following condition holds $\sup_{c\in\tilde{S}}|c| \leq \kappa\sigma$, we have*

$$\|[PA, L_\tau]\| \leq C_1\|\nabla\tau\|_\infty,$$

with the same constant C_1 as in Lemma 2.5, which depends on h and κ. Similarly, we have

$$\|L_\tau A - A\| \leq \frac{C_2}{\sigma}\|\tau\|_\infty,$$

with C_2 as defined in Lemma 2.6.

By constructing a multi-layer representation $\Phi_n(x)$ in $L^2(G)$ using similar operators at each layer, we can obtain a similar stability result to Theorem 2.7. By adding a global pooling operator $A_c : L^2(G) \to L^2(\mathbb{R}^2)$ after the last layer, defined, for $x \in L^2(G)$, as

$$A_c x(u) = \int x((u,\eta))d\mu_c(\eta),$$

we additionally obtain global invariance to rotations, as shown in the following theorem.

Theorem 2.10 (Stability and global rotation invariance)**.** *Assume* (A2) *for patches $\tilde{S}$ at each layer. Define the operator $L_{(\tau,\theta)}x((u,\eta)) = x((\tau(u),\theta)^{-1}(u,\eta))$, and define the diffeomorphism $\tau_\theta : u \mapsto R_{-\theta}\tau(u)$. If $\|\nabla\tau\|_\infty \leq 1/2$, we have*

$$\begin{aligned}\|A_c\Phi_n(L_{(\tau,\theta)}x) - A_c\Phi_n(x)\| &\leq \|\Phi_n(L_{R_{\tau_\theta}}x) - \Phi_n(x)\| \\ &\leq \left(C_1\,(1+n)\,\|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n}\|\tau\|_\infty\right)\|x\|.\end{aligned}$$

We note that a similar result may be obtained when $G = \mathbb{R}^d \rtimes H$, where H is any compact group, with a possible additional dependence on how elements of H affect the size of patches.

2.4 Link with Existing Convolutional Architectures

In this section, we study the functional spaces (RKHS) that arise from our multilayer kernel representation, and examine the connections with more standard convolutional architectures. The motivation of this study is that if a CNN model f is in the RKHS, then it can be written in a "linearized" form $f(x) = \langle f, \Phi(x)\rangle$, so that our study of stability of the kernel representation Φ extends to predictions using $|f(x) - f(x')| \leq \|f\|\|\Phi(x) - \Phi(x')\|$.

We begin by considering in Section 2.4.1 the intermediate kernels K_k, showing that their RKHSs contain simple neural-network-like functions defined on patches with smooth activations, while in Section 2.4.2 we show that a certain class of generic CNNs are contained in the RKHS $\mathcal{H}_{\mathcal{K}_n}$ of the full multilayer kernel $\mathcal{K}_n$ and characterize their norm. This is achieved by considering particular functions in each intermediate RKHS defined in terms of the convolutional filters of the CNN. A consequence of these results is that our stability and invariance properties from Section 2.3 are valid for this broad class of CNNs.

2.4.1 Activation Functions and Kernels K_k

Before introducing formal links between our kernel representation and classical convolutional architectures, we study in more details the kernels K_k described in Section 2.2 and their RKHSs $\mathcal{H}_k$. In particular, we are interested in characterizing which types of functions live in $\mathcal{H}_k$. The next lemma extends some results of Zhang et al. (2016, 2017b), originally developed for the inverse polynomial and Gaussian kernels; it shows that the RKHS may contain simple "neural network" functions with activations σ that are smooth enough.

Lemma 2.11 (Activation functions and RKHSs $\mathcal{H}_k$)**.** *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a function that admits a polynomial expansion $\sigma(u) := \sum_{j=0}^{\infty} a_j u^j$. Consider a kernel K_k from Section 2.2, given in (2.2), with $\kappa_k(u) = \sum_{j=0}^{\infty} b_j u^j$, and $b_j \geq 0$ for all j. Assume further that $a_j = 0$ whenever $b_j = 0$, and define the function $C_\sigma^2(\lambda^2) := \sum_{j=0}^{\infty}(a_j^2/b_j)\lambda^{2j}$. Let g in $\mathcal{P}_k$ be such that $C_\sigma^2(\|g\|^2) < \infty$. Then, the RKHS $\mathcal{H}_k$ contains the function*

$$f : z \mapsto \|z\|\sigma(\langle g, z\rangle/\|z\|), \tag{2.15}$$

and its norm satisfies $\|f\| \leq C_\sigma(\|g\|^2)$.

Noting that for all examples of κ_k given in Section 2.2, we have $b_1 > 0$, this result implies the next corollary, which was also found to be useful in our analysis.

Corollary 2.12 (Linear functions and RKHSs)**.** *The RKHSs $\mathcal{H}_k$ for the examples of κ_k given in Section 2.2 contain all linear functions of the form $z \mapsto \langle g, z\rangle$ with g in $\mathcal{P}_k$.*

The previous lemma shows that for many choices of smooth functions σ, the RKHS $\mathcal{H}_k$ contains the functions of the form (2.15). While the non-homogeneous functions $z \mapsto \sigma(\langle g, z\rangle)$ are standard in neural networks, the homogeneous variant is not. Yet, we note that (i) the most successful activation function, namely rectified linear units, is homogeneous—that is, $\mathrm{relu}(\langle g, z\rangle) = \|z\|\mathrm{relu}(\langle g, z\rangle/\|z\|)$; (ii) while relu is nonsmooth and thus not in our RKHSs, there exists a smoothed variant that satisfies the conditions of Lemma 2.11 for useful kernels. As noticed by Zhang et al. (2016, 2017b), this is for instance the case for the inverse polynomial kernel. In Figure 2.4, we plot and compare these different variants of ReLU.

2.4.2 Convolutional Neural Networks and their Complexity

We now study the connection between the kernel representation defined in Section 2.2 and CNNs. Specifically, we show that the RKHS of the final kernel $\mathcal{K}_n$ obtained from our kernel construction contains a set of CNNs on continuous domains with certain types of smooth homogeneous activations. An important consequence is that the stability results of previous sections apply to this class of CNNs, although the stability depends on the RKHS norm, as discussed later in Section 2.5. This norm also serves as a measure of model complexity, thus controlling both generalization and stability.

CNN maps construction. We now define a CNN function f_σ that takes as input an image z_0 in $L^2(\mathbb{R}^d, \mathbb{R}^{p_0})$ with p_0 channels, and build a sequence of feature maps, represented at layer k as a function z_k in $L^2(\mathbb{R}^d, \mathbb{R}^{p_k})$ with p_k channels; the map z_k is

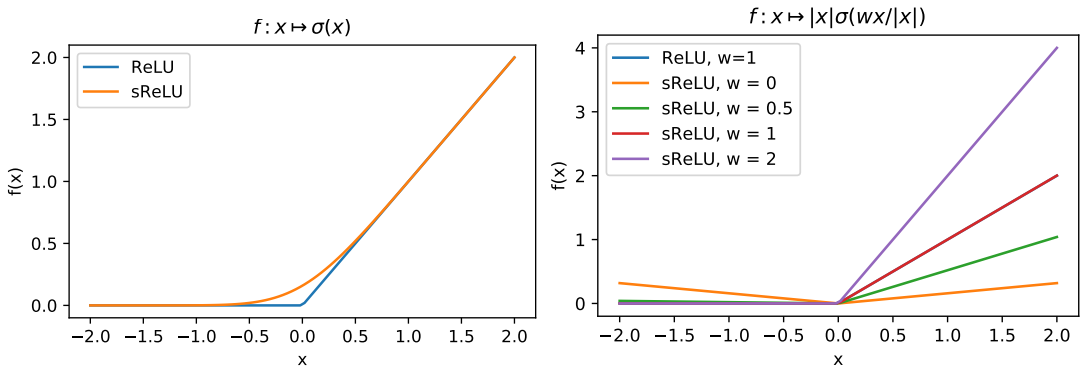


Figure 2.4: Comparison of one-dimensional functions obtained with relu and smoothed relu (sReLU) activations. (Left) non-homogeneous setting of Zhang et al. (2016, 2017b). (Right) our homogeneous setting, for different values of the parameter w. Note that for $w \geq 0.5$, sReLU and ReLU are indistinguishable.

obtained from z_{k-1} by performing linear convolutions with a set of filters $(w_k^i)_{i=1,\dots,p_k}$, followed by a pointwise activation function σ to obtain an intermediate feature map $\tilde{z}_k$, then by applying a linear pooling filter. Note that each w_k^i is in $L^2(S_k, \mathbb{R}^{p_{k-1}})$, with channels denoted by w_k^{ij} in $L^2(S_k, \mathbb{R})$. Formally, the intermediate map $\tilde{z}_k$ in $L^2(\mathbb{R}^d, \mathbb{R}^{p_k})$ is obtained by

$$\tilde{z}_k^i(u) = n_k(u)\sigma\Big(\langle w_k^i, P_k z_{k-1}(u)\rangle / n_k(u)\Big)\,, \tag{2.16}$$

where $\tilde{z}_k(u) = (\tilde{z}_k^1(u), \dots, \tilde{z}_k^{p_k}(u))$ is in $\mathbb{R}^{p_k}$, and P_k is a patch extraction operator for finite-dimensional maps. The activation involves a pointwise non-linearity σ along with a quantity $n_k(u) := \|P_k x_{k-1}(u)\|$ in (2.16), which is due to the homogenization, and which is independent of the filters w_k^i. Finally, the map z_k is obtained by using a pooling operator as in Section 2.2, with $z_k = A_k \tilde{z}_k$, and $z_0 = x_0$.

Prediction layer. For simplicity, we consider the case of a linear fully connected prediction layer. In this case, the final CNN prediction function f_σ is given by

$$f_\sigma(x_0) = \langle w_{n+1}, z_n \rangle,$$

with parameters w_{n+1} in $L^2(\mathbb{R}^d, \mathbb{R}^{p_n})$. We now show that such a CNN function is contained in the RKHS of the kernel $\mathcal{K}_n$ defined in (2.6).

Construction in the RKHS. The function f_σ can be constructed recursively from intermediate functions that lie in the RKHSs $\mathcal{H}_k$, of the form (2.15), for appropriate activations σ. Specifically, we define initial quantities f_1^i in $\mathcal{H}_1$ and g_1^i in $\mathcal{P}_1$ for $i = 1, \dots, p_1$ such that

$$\begin{aligned} g_1^i &= w_1^i \in L^2(S_1, \mathbb{R}^{p_0}) = L^2(S_1, \mathcal{H}_0) = \mathcal{P}_1, \\ f_1^i(z) &= \|z\|\sigma(\langle g_i^0, z\rangle / \|z\|) \quad \text{for } z \in \mathcal{P}_1, \end{aligned}$$

and we define, from layer k–1, the quantities f_k^i in $\mathcal{H}_k$ and g_k^i in $\mathcal{P}_k$ for $i = 1, \ldots, p_k$:

$$g_k^i(v) = \sum_{j=1}^{p_{k-1}} w_k^{ij}(v) f_{k-1}^j \quad \text{where} \quad w_k^i(v) = (w_k^{ij}(v))_{j=1,\ldots,p_{k-1}},$$
$$f_k^i(z) = \|z\| \sigma(\langle g_k^i, z\rangle / \|z\|) \quad \text{for } z \in \mathcal{P}_k.$$

For the linear prediction layer, we define g_σ in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ by:

$$g_\sigma(u) = \sum_{j=1}^{p_n} w_{n+1}^j(u) f_n^j \quad \text{for all } u \in \mathbb{R}^d,$$

so that the function $f : x_0 \mapsto \langle g_\sigma, x_n \rangle$ is in the RKHS of $\mathcal{K}_n$, where x_n is the final representation given in Eq. (2.5). In Appendix 2.D.2, we show that $f = f_\sigma$, which implies that the CNN function f_σ is in the RKHS. We note that a similar construction for fully connected multilayer networks with constraints on weights and inputs was given by Zhang et al. (2016).

Norm of the CNN f_σ. We now study the RKHS norm of the CNN constructed above. This quantity is important as it controls the stability and invariance of the predictions of a learned model through (2.1). Additionally, the RKHS norm provides a way to control model complexity, and can lead to generalization bounds, *e.g.*, through Rademacher complexity and margin bounds (Boucheron et al., 2005; Shalev-Shwartz and Ben-David, 2014). In particular, such results rely on the following upper bound on the empirical Rademacher complexity of a function class with bounded RKHS norm $\mathcal{F}_\lambda = \{f \in \mathcal{H}_{\mathcal{K}_n} : \|f\| \leq \lambda\}$, for a dataset $\{x^{(1)}, \ldots, x^{(N)}\}$:

$$R_N(\mathcal{F}_\lambda) \leq \frac{\lambda \sqrt{\frac{1}{N}\sum_{i=1}^N \mathcal{K}_n(x^{(i)}, x^{(i)})}}{\sqrt{N}}. \tag{2.17}$$

The bound remains valid when only considering CNN functions in $\mathcal{F}_\lambda$ of the form f_σ, since such a function class is contained in $\mathcal{F}_\lambda$. If we consider a binary classification task with training labels $y^{(i)}$ in $\{-1, 1\}$, on can then obtain a margin-based bound for any function f_N in $\mathcal{F}_\lambda$ obtained from the training set and any margin $\gamma > 0$: with probability $1 - \delta$, we have (see, *e.g.*, Boucheron et al., 2005)

$$L(f_N) \leq L_N^\gamma(f_N) + O\left(\frac{\lambda \sqrt{\frac{1}{N}\sum_{i=1}^N \mathcal{K}_n(x^{(i)}, x^{(i)})}}{\gamma \sqrt{N}} + \sqrt{\frac{\log(1/\delta)}{N}} \right), \tag{2.18}$$

with

$$L(f) = \mathbb{P}_{(x,y)\sim\mathcal{D}}(yf(x) < 0)$$
$$L_N^\gamma(f) = \frac{1}{N}\sum_{i=1}^N \mathbb{1}\{y^{(i)} f(x^{(i)}) < \gamma\},$$

where $\mathcal{D}$ is the distribution of data-label pairs $(x^{(i)}, y^{(i)})$. Intuitively, the margin γ corresponds to a level of confidence, and L_N^γ measures training error when requiring

confident predictions. Then, the bound on the gap between this training error and the true expected error $L(f_N)$ becomes larger for small confidence levels, and is controlled by the model complexity λ and the sample size N.

Note that the bound requires a fixed value of λ used during training, but in practice, learning under a constraint $\|f\| \leq \lambda$ can be difficult, especially for CNNs which are typically trained with stochastic gradient descent with little regularization. However, by considering values of λ on a logarithmic scale and taking a union bound, one can obtain a similar bound with $\|f_N\|$ instead of λ, up to logarithmic factors (see, *e.g.*, Shalev-Shwartz and Ben-David, 2014, Theorem 26.14), where f_N is obtained from the training data. We note that various authors have recently considered other norm-based complexity measures to control the generalization of neural networks with more standard activations (see, *e.g.*, Bartlett et al., 2017; Liang et al., 2018; Neyshabur et al., 2017, 2015a). However, their results are typically obtained for fully connected networks on finite-dimensional inputs, while we consider CNNs for input signals defined on continuous domains. The next proposition (proved in Appendix 2.D.2) characterizes the norm of f_σ in terms of the L^2 norms of the filters w_k^{ij}, and follows from the recursive definition of the intermediate RKHS elements f_k^i.

Proposition 2.13 (RKHS norm of CNNs)**.** *Assume the activation σ satisfies $C_\sigma(a) < \infty$ for all $a \geq 0$, where C_σ is defined for a given kernel in Lemma 2.11. Then, the CNN function f_σ defined above is in the RKHS $\mathcal{H}_{\mathcal{K}_n}$, with norm*

$$\|f_\sigma\|^2 \leq p_n \sum_{i=1}^{p_n} \|w_{n+1}^i\|_2^2 B_{n,i},$$

where $B_{n,i}$ is defined by $B_{1,i} = C_\sigma^2(\|w_1^i\|_2^2)$ and $B_{k,i} = C_\sigma^2\left(p_{k-1} \sum_{j=1}^{p_{k-1}} \|w_k^{ij}\|_2^2 B_{k-1,j}\right)$.

Note that this upper bound need not grow exponentially with depth when the filters have small norm and C_σ takes small values around zero. However, the dependency of the bound on the number of feature maps p_k of each layer k may not be satisfactory in situations where the number of parameters is very large, which is common in successful deep learning architectures. The following proposition removes this dependence, relying instead on matrix spectral norms. Similar quantities have been used recently to obtain useful generalization bounds for neural networks (Bartlett et al., 2017; Neyshabur et al., 2018).

Proposition 2.14 (RKHS norm of CNNs using spectral norms)**.** *Assume the activation σ satisfies $C_\sigma(a) < \infty$ for all $a \geq 0$, where C_σ is defined for a given kernel in Lemma 2.11. Then, the CNN function f_σ defined above is in the RKHS $\mathcal{H}_{\mathcal{K}_n}$, with norm*

$$\|f_\sigma\|^2 \leq \|w_{n+1}\|^2 \; C_\sigma^2(\|W_n\|_2^2 \; C_\sigma^2(\|W_{n-1}\|_2^2 \ldots C_\sigma^2(\|W_2\|_2^2 \; C_\sigma^2(\|W_1\|_F^2)) \ldots)). \tag{2.19}$$

The norms are defined as follows:

$$\|W_k\|_2^2 = \int_{S_k} \|W_k(u)\|_2^2 d\nu_k(u), \quad \text{for } k = 2, \ldots, n$$
$$\|W_1\|_F^2 = \int_{S_1} \|W_1(u)\|_F^2 d\nu_1(u),$$

where $W_k(u)$ is the matrix $(w_k^{ij}(u))_{ij}$, $\|\cdot\|_2$ the spectral norm, and $\|\cdot\|_F$ the Frobenius norm.

As an example, if we consider $\kappa_1 = \cdots = \kappa_n$ to be one of the kernels introduced in Section 2.2 and take $\sigma = \kappa_1$ so that $C_\sigma^2(\lambda^2) = \kappa_1(\lambda^2)$, then constraining the norms at each layer to be smaller than 1 ensures $\|f_\sigma\| \leq 1$, since for $\lambda \leq 1$ we have $C_\sigma^2(\lambda^2) \leq C_\sigma^2(1) = \kappa_1(1) = 1$. If we consider linear kernels and $\sigma(u) = u$, we have $C_\sigma^2(\lambda^2) = \lambda^2$ and the bound becomes $\|f_\sigma\| \leq \|w_{n+1}\|\|W_n\|_2 \cdots \|W_2\|_2\|W_1\|_F$. If we ignore the convolutional structure (*i.e.*, only taking 1x1 patches on a 1x1 image), the norm involves a product of spectral norms at each layer (ignoring the first layer), a quantity which also appears in recent generalization bounds (Bartlett et al., 2017; Neyshabur et al., 2018). While such quantities have proven useful to explain some generalization phenomena, such as the behavior of networks trained on data with random labels (Bartlett et al., 2017; Zhang et al., 2017a), some authors have pointed out that spectral norms may yield overly pessimistic generalization bounds when comparing with simple parameter counting (Arora et al., 2018), and our results may display similar drawbacks. We note, however, that Proposition 2.14 only gives an upper bound, and the actual RKHS norm may be smaller in practice. It may also be that the norm is not well controlled during training, and that the obtained bounds may not fully explain the generalization behavior observed in practice. Using such quantities to regularize during training (as in Chapter 3) may then yield bounds that are less vacuous.

Generalization and stability. The results of this section imply that our study of the geometry of the kernel representations, and in particular the stability and invariance properties of Section 2.3, apply to the generic CNNs defined above, thanks to the Lipschitz smoothness relation (2.1). The smoothness is then controlled by the RKHS norm of these functions, which sheds light on the links between generalization and stability. In particular, functions with low RKHS norm provide better generalization guarantees on unseen data, as shown by the margin bound in Eq. (2.18). This implies, for instance, that generalization is harder if the task requires classifying two slightly deformed images with different labels, since separating such predictions by some margin requires a function with large RKHS norm according to our stability analysis. In contrast, if a stable function (*i.e.*, with small RKHS norm) is sufficient to do well on a training set, learning becomes "easier" and few samples may be enough for good generalization.

2.4.3 Stability and Generalization with Generic Activations

Our study of stability and generalization so far has relied on kernel methods, which allows us to separate learned models from data representations in order to establish tight connections between the stability of representations and statistical properties of learned CNNs through RKHS norms. One important caveat, however, is that our study is limited to CNNs with a class of smooth and homogeneous activations described in Section 2.4.1, which differ from generic activations used in practice such as ReLU or tanh. Indeed, ReLU is homogeneous but lacks the required smoothness, while tanh is not homogeneous. In this section, we show that our stability results can be extended to the predictions of CNNs with such activations, and that stability is controlled by a quantity based on spectral norms, which plays an important role in recent results on generalization. This confirms a strong connection between stability and generalization in this more general context as well.

Stability bound. We consider an activation function $\sigma : \mathbb{R} \to \mathbb{R}$ that is ρ-Lipschitz and satisfies $\sigma(0) = 0$. Examples include ReLU and tanh activations, for which $\rho = 1$. The CNN construction is similar to Section 2.4.2 with feature maps z_k in $L^2(\mathbb{R}^d, \mathbb{R}^{p_k})$, and a final prediction function f_σ defined with an inner product $\langle w_{n+1}, z_n \rangle$. The only change is the non-linear mapping in Eq. (2.16), which is no longer homogeneized, and can be rewritten as

$$\tilde{z}_k(u) = \varphi_k(P_k z_{k-1}(u)) := \sigma\Big(\int_{S_k} W_k(v)(P_k z_{k-1}(u))(v) d\nu_k(v)\Big),$$

where σ is applied component-wise. The non-linear mapping φ_k on patches satisfies

$$\|\varphi_k(z) - \varphi_k(z')\| \leq \rho_k \|z - z'\| \quad \text{and} \quad \|\varphi_k(z)\| \leq \rho_k \|z\|,$$

where $\rho_k = \rho \|W_k\|_2$ and $\|W_k\|_2$ is the spectral norm of $W_k : L^2(S_k, \mathbb{R}^{p_{k-1}}) \to \mathbb{R}^{p_k}$ defined by

$$\|W_k\|_2 = \sup_{\|z\| \leq 1} \left\| \int_{S_k} W_k(v) z(v) d\nu_k(v) \right\|.$$

We note that this spectral norm is slightly different than the mixed norm used in Proposition 2.14. By defining an operator M_k that applies φ_k pointwise as in Section 2.2, the construction of the last feature map takes the same form as that of the multilayer kernel representation, so that the results of Section 2.3 apply, leading to the following stability bound on the final predictions:

$$|f_\sigma(L_\tau x) - f_\sigma(x)| \leq \rho^n \|w_{n+1}\| \left(\prod_k \|W_k\|_2 \right) \left(C_1 (1+n) \|\nabla \tau\|_\infty + \frac{C_2}{\sigma_n} \|\tau\|_\infty \right) \|x\|. \tag{2.20}$$

Link with generalization. The stability bound (2.20) takes a similar form to the one obtained for CNNs in the RKHS, with the RKHS norm replaced by the product of spectral norms. In contrast to the RKHS norm, such a quantity does not directly lead to generalization bounds; however, a few recent works have provided meaningful generalization bounds for deep neural networks that involve the product of spectral norms (Bartlett et al., 2017; Neyshabur et al., 2018). Thus, this suggests that stable CNNs have better generalization properties, even when considering generic CNNs with ReLU or tanh activations. Nevertheless, these bounds typically involve an additional factor consisting of other matrix norms summed across layers, which may introduce some dependence on the number of parameters, and do not directly support convolutional structure. In contrast, our RKHS norm bound based on spectral norms given in Proposition 2.14 directly supports convolutional structure, and has no dependence on the number of parameters.

2.5 Discussion and Concluding Remarks

In this chapter, we introduce a multilayer convolutional kernel representation (Section 2.2); we show that it is stable to the action of diffeomorphisms, and that it can be made invariant to groups of transformations (Section 2.3); and finally we explain

connections between our representation and generic convolutional networks by showing that certain classes of CNNs with smooth activations are contained in the RKHS of the full multilayer kernel (Section 2.4). A consequence of this last result is that the stability results of Section 2.3 apply to any CNN function f from that class, by using the relation

$$|f(L_\tau x) - f(x)| \leq \|f\| \|\Phi_n(L_\tau x) - \Phi_n(x)\|,$$

which follows from (2.1), assuming a linear prediction layer. In the case of CNNs with generic activations such as ReLU, the kernel point of view is not applicable, and the separation between model and representation is not as clear. However, we show in Section 2.4.3 that a similar stability bound can be obtained, with the product of spectral norms at each layer playing a similar role to the RKHS norm of the CNN. In both cases, a quantity that characterizes complexity of a model appears in the final bound on predicted values — either the RKHS norm or the product of spectral norms —, and this complexity measure is also closely related to generalization. This implies that learning with stable CNNs is "easier" in terms of sample complexity, and that the inductive bias of CNNs is thus suitable to tasks that present some invariance under translation and small local deformation, as well as more general transformation groups, when the architecture is appropriately constructed.

In order to ensure stability, the previous bounds suggest that one should control the RKHS norm $\|f\|$, or the product of spectral norms when using generic activations; however, these quantities are difficult to control with standard approaches to learning CNNs, such as backpropagation. In contrast, traditional kernel methods typically control this norm by using it as an explicit regularizer in the learning process, making such a stability guarantee more useful. In order to avoid the scalability issues of such approaches, convolutional kernel networks approximate the full kernel map Φ_n by taking appropriate projections as explained in Section 2.2.2, leading to a representation $\tilde{\Phi}_n$ that can be represented with a practical representation ψ_n that preserves the Hilbert space structure isometrically (using the finite-dimensional descriptions of points in the RKHS given in (2.10)). Section 2.3.3 shows that such representations satisfy the same stability and invariance results as the full representation, at the cost of losing information. Then, if we consider a CKN function of the form $f_w(x) = \langle w, \psi_n(x) \rangle$, stability is obtained thanks to the relation

$$|f_w(L_\tau x) - f_w(x)| \leq \|w\| \|\psi_n(L_\tau x) - \psi_n(x)\| = \|w\| \|\tilde{\Phi}_n(L_\tau x) - \tilde{\Phi}_n(x)\|.$$

In particular, learning such a function by controlling the norm of w, *e.g.*, with ℓ_2 regularization, provides a natural way to explicitly control stability.

Implicit and explicit regularization of CNNs. In the context of generic CNNs trained with backpropagation and variants of stochastic gradient descent, it has been suggested (see, *e.g.*, Zhang et al., 2017a) that optimization algorithms play an important role in controlling their generalization ability. Viewing a generic CNN with ReLU activations as approximately an element of the RKHS as described in Section 2.4.2 (up to the smoothed activations), we may then wonder if training algorithms implicitly control the RKHS norm for a kernel as defined in this chapter. Yet, modern CNNs trained with SGD have been found to be highly unstable to small, additive perturbations known as "adversarial examples" (Szegedy et al., 2014), which suggests that the RKHS norm of

these models may be quite large, and that controlling it explicitly during learning might be important to learn more stable models—this is the subject of Chapter 3 in this thesis.

Separately, a recent line of work has studied relationships between gradient training of neural network in a certain over-parameterized regime and kernel methods, through so-called *neural tangent kernels* (Jacot et al., 2018; Allen-Zhu et al., 2019b; Du et al., 2019b; Chizat et al., 2019). In this case, optimization may implicitly bias learning towards functions with small norm in a RKHS, however learning is done *improperly*, that is, one may learn stable target functions in the RKHS, but the networks used for learning are *not* in the RKHS themselves, as in the case of random feature approximations (Rahimi and Recht, 2007; Bach, 2017b), which may lead to good generalization despite poor stability properties of the neural network. The inductive bias of the kernels arising in such a regime is the subject of Chapter 4.

Appendix

2.A Useful Mathematical Tools

In this section, we present preliminary mathematical tools that are used in our analysis.

Harmonic analysis. We recall a classical result from harmonic analysis (see, *e.g.*, Stein, 1993), which was used many times by Mallat (2012) to prove the stability of the scattering transform to the action of diffeomorphisms.

Lemma 2.A.1 (Schur's test)**.** *Let $\mathcal{H}$ be a Hilbert space and Ω a subset of $\mathbb{R}^d$. Consider T an integral operator with kernel $k : \Omega \times \Omega \to \mathbb{R}$, meaning that for all u in Ω and x in $L^2(\Omega, \mathcal{H})$,*

$$Tx(u) = \int_\Omega k(u, v)x(v)dv, \tag{2.21}$$

where the integral is a Bochner integral (see, Diestel and Uhl, 1977; Muandet et al., 2017) when $\mathcal{H}$ is infinite-dimensional. If

$$\forall u \in \Omega, \quad \int |k(u, v)|dv \leq C \quad \text{and} \quad \forall v \in \Omega, \quad \int |k(u, v)|du \leq C,$$

for some constant C, then, Tx is always in $L^2(\Omega, \mathcal{H})$ for all x in $L^2(\Omega, \mathcal{H})$ and we have $\|T\| \leq C$.

Note that while the proofs of the lemma above are typically given for real-valued functions in $L^2(\Omega, \mathbb{R})$, the result can easily be extended to Hilbert space-valued functions x in $L^2(\Omega, \mathcal{H})$. In order to prove this, we consider the integral operator $|T|$ with kernel $|k|$ that operates on $L^2(\Omega, \mathbb{R}_+)$, meaning that $|T|$ is defined as in (2.21) by replacing $k(u, v)$ by the absolute value $|k(u, v)|$. Then, consider x in $L^2(\Omega, \mathcal{H})$ and use the triangle inequality property of Bochner integrals:

$$\|Tx\|^2 = \int_\Omega \|Tx(u)\|^2 du \leq \int_\Omega \left(\int_\Omega |k(u, v)| \|x(v)\| dv \right)^2 du = \||T||x|\|^2,$$

where the function $|x|$ is such that $|x|(u) = \|x(u)\|$ and thus $|x|$ is in $L^2(\Omega, \mathbb{R}_+)$. We may now apply Schur's test to the operator $|T|$ for real-valued functions, which gives $\||T|\| \leq C$. Then, noting that $\||x|\| = \|x\|$, we conclude with the inequality $\|Tx\|^2 \leq \||T||x|\|^2 \leq \||T|\|^2 \|x\|^2 \leq C^2 \|x\|^2$.

The following lemma shows that the pooling operators A_k defined in Section 2.2 are non-expansive.

Lemma 2.A.2 (Non-expansiveness of pooling operators)**.** *If* $h(u) := (2\pi)^{-d/2}\exp(-|u|^2/2)$*, then the pooling operator* A_σ *defined for any* $\sigma > 0$ *by*

$$A_\sigma x(u) = \int_{\mathbb{R}^d} \sigma^{-d} h\left(\frac{u-v}{\sigma}\right) x(v)dv,$$

has operator norm $\|A_\sigma\| \leq 1$*.*

Proof. With the notations from above, we have $\|A_\sigma x\| \leq \||A_\sigma||x|\| = \|h_\sigma * |x|\|$, where $h_\sigma := \sigma^{-d}h(\cdot/\sigma)$ and $*$ denotes convolution. By Young's inequality, we have $\|h_\sigma * |x|\| \leq \|h_\sigma\|_1 \cdot \||x|\| = 1 \cdot \||x|\| = \|x\|$, which concludes the proof. □

Kernel methods. We now recall a classical result that characterizes the reproducing kernel Hilbert space (RKHS) of functions defined from explicit Hilbert space mappings (see, *e.g.*, Saitoh, 1997, §2.1).

Theorem 2.A.1. *Let* $\psi : \mathcal{X} \to H$ *be a feature map to a Hilbert space* H*, and let* $K(z, z') := \langle \psi(z), \psi(z')\rangle_H$ *for* $z, z' \in \mathcal{X}$*. Let* $\mathcal{H}$ *be the linear subspace defined by*

$$\mathcal{H} := \{f_w \ ; \ w \in H\} \quad s.t. \quad f_w : z \mapsto \langle w, \psi(z)\rangle_H,$$

and consider the norm

$$\|f_w\|_{\mathcal{H}}^2 := \inf_{w' \in H} \{\|w'\|_H^2 \quad s.t. \quad f_w = f_{w'}\}.$$

Then $\mathcal{H}$ *is the reproducing kernel Hilbert space associated to kernel* K*.*

A consequence of this result is that the RKHS of the kernel $\mathcal{K}_n(x, x') = \langle \Phi(x), \Phi(x')\rangle$, defined from a given final representation $\Phi(x) \in \mathcal{H}_{n+1}$ such as the one introduced in Section 2.2, contains functions of the form $f : x \mapsto \langle w, \Phi(x)\rangle$ with $w \in \mathcal{H}_{n+1}$, and the RKHS norm of such a function satisfies $\|f\| \leq \|w\|_{\mathcal{H}_{n+1}}$.

2.B Proofs Related to the Multilayer Kernel Construction

2.B.1 Proof of Lemma 2.1 and Non-Expansiveness of the Gaussian Kernel

We begin with the proof of Lemma 2.1 related to homogeneous dot-product kernels (2.2).

Proof. In this proof, we drop all indices k since there is no ambiguity. We will prove the more general result that φ is ρ_k-Lipschitz with $\rho_k = \max(1, \sqrt{\kappa'(1)})$ for any value of $\kappa'(1)$ (in particular, it is non-expansive when $\kappa'(1) \leq 1$).

Let us consider the Maclaurin expansion $\kappa(u) = \sum_{j=0}^{+\infty} b_j u^j < +\infty$ with $b_j \geq 0$ for all j and all u in $[-1, +1]$. Recall that the condition $b_j \geq 0$ comes from the positive-definiteness of K (Schoenberg, 1942). Then, we have $\kappa'(u) = \sum_{j=1}^{+\infty} j b_j u^{j-1}$. Noting that $j b_j u^{j-1} \leq j b_j$ for $u \in [-1, 1]$, we have $\kappa'(u) \leq \kappa'(1)$ on $[-1, 1]$. The fundamental theorem of calculus then yields, for $u \in [-1, 1]$,

$$\kappa(u) = \kappa(1) - \int_u^1 \kappa'(t)dt \geq \kappa(1) - \kappa'(1)(1-u). \tag{2.22}$$

Then, if $z, z' \neq 0$,

$$\|\varphi(z) - \varphi(z')\|^2 = K(z,z) + K(z',z') - 2K(z,z') = \|z\|^2 + \|z'\|^2 - 2\|z\|\|z'\|\kappa(u),$$

with $u = \langle z, z' \rangle / (\|z\|\|z'\|)$. Using (2.22) with $\kappa(1) = 1$, we have

$$\begin{aligned}
\|\varphi(z) - \varphi(z')\|^2 &\leq \|z\|^2 + \|z'\|^2 - 2\|z\|\|z'\| \left(1 - \kappa'(1) + \kappa'(1)u\right) \\
&= (1 - \kappa'(1)) \left(\|z\|^2 + \|z'\|^2 - 2\|z\|\|z'\|\right) + \kappa'(1) \left(\|z\|^2 + \|z'\|^2 - 2\langle z, z' \rangle\right) \\
&= (1 - \kappa'(1)) \left|\|z\| - \|z'\|\right|^2 + \kappa'(1)\|z - z'\|^2 \\
&\leq \begin{cases} \|z - z'\|^2, & \text{if } 0 \leq \kappa'(1) \leq 1 \\ \kappa'(1)\|z - z'\|^2, & \text{if } \kappa'(1) > 1 \end{cases} \\
&= \rho_k^2 \|z - z'\|^2,
\end{aligned}$$

with $\rho_k = \max(1, \sqrt{\kappa'(1)})$, which yields the desired result. Finally, we remark that we have shown the relation $\kappa(u) \geq \kappa(1) - \kappa'(1) + \kappa'(1)u$; when $\kappa'(1) = 1$, this immediately yields (2.3).

If $z = 0$ or $z' = 0$, the result also holds trivially. For example,

$$\|\varphi(z) - \varphi(0)\|^2 = K(z,z) + K(0,0) - 2K(z,0) = \|z\|^2 = \|z - 0\|^2.$$

□

Non-expansiveness of the Gaussian kernel. We now consider the Gaussian kernel

$$K(z,z') := e^{-\frac{\alpha}{2}\|z - z'\|^2},$$

with feature map φ. We simply use the convexity inequality $e^u \geq 1 + u$ for all u, and

$$\|\varphi(z) - \varphi(z')\|^2 = K(z,z) + K(z',z') - 2K(z,z') = 2 - 2e^{-\frac{\alpha}{2}\|z - z'\|^2} \leq \alpha\|z - z'\|^2.$$

In particular, φ is non-expansive when $\alpha \leq 1$.

2.C Proofs of Recovery and Stability Results

2.C.1 Proof of Lemma 2.2

Proof. We denote by $\bar{\Omega}$ the discrete set of sampling points considered in this lemma. The assumption on $\bar{\Omega}$ can be written as $\{u + v \ ; \ u \in \bar{\Omega}, v \in S_k\} = \mathbb{R}^d$.

Let B denote an orthonormal basis of the Hilbert space $\mathcal{P}_k = L^2(S_k, \mathcal{H}_{k-1})$, and define the linear function f_w in $\mathcal{H}_k$ such that $f_w : z \mapsto \langle w, z \rangle$ for w in $\mathcal{P}_k$. We thus have

$$\begin{aligned}
P_k x_{k-1}(u) &= \sum_{w \in B} \langle w, P_k x_{k-1}(u) \rangle w \\
&= \sum_{w \in B} f_w(P_k x_{k-1}(u)) w \\
&= \sum_{w \in B} \langle f_w, M_k P_k x_{k-1}(u) \rangle w,
\end{aligned}$$

using the reproducing property in the RKHS $\mathcal{H}_k$. Applying the pooling operator A_k yields

$$\begin{aligned} A_k P_k x_{k-1}(u) &= \sum_{w \in B} \langle f_w, A_k M_k P_k x_{k-1}(u) \rangle w, \\ &= \sum_{w \in B} \langle f_w, x_k(u) \rangle w. \end{aligned}$$

Noting that $A_k P_k x_{k-1} = A_k (L_v x_{k-1})_{v \in S_k} = (A_k L_v x_{k-1})_{v \in S_k} = (L_v A_k x_{k-1})_{v \in S_k} = P_k A_k x_{k-1}$, with $L_v x_{k-1}(u) := x_{k-1}(u+v)$, we can choose v in S_k and obtain from the previous relations

$$A_k x_{k-1}(u+v) = \sum_{w \in B} \langle f_w, x_k(u) \rangle w(v).$$

Thus, taking all sampling points $u \in \bar{\Omega}$ and all $v \in S_k$, we have a full view of the signal $A_k x_{k-1}$ on all of $\mathbb{R}^d$ by our assumption on the set $\bar{\Omega}$.

For $f \in \mathcal{H}_{k-1}$, the signal $\langle f, x_{k-1}(u) \rangle$ can then be recovered by deconvolution as follows:

$$\langle f, x_{k-1}(u) \rangle = \mathcal{F}^{-1} \left(\frac{\mathcal{F}(\langle f, A_k x_{k-1}(\cdot) \rangle)}{\mathcal{F}(h_{\sigma_k})} \right)(u),$$

where $\mathcal{F}$ denotes the Fourier transform. Note that the inverse Fourier transform is well-defined here because the signal $\langle f, A_k x_k(\cdot) \rangle$ is itself a convolution with h_{σ_k}, and $\mathcal{F}(h_{\sigma_k})$ is strictly positive as the Fourier transform of a Gaussian is also a Gaussian.

By considering all elements f in an orthonormal basis of $\mathcal{H}_{k-1}$, we can recover x_{k-1}. The map x_k can then be reconstructed trivially by applying operators P_k, M_k and A_k on x_{k-1}.

□

2.C.2 Proof of Lemma 2.3

Proof. In this proof, we drop the bar notation on all quantities for simplicity; there is indeed no ambiguity since all signals are discrete here. First, we recall that $\mathcal{H}_k$ contains all linear functions on $\mathcal{P}_k = \mathcal{H}_{k-1}^{e_k}$; thus, we may consider in particular functions $f_{j,w}(z) := e_k^{1/2} \langle w, z_j \rangle$ for $j \in \{1, \dots, e_k\}$, $w \in \mathcal{H}_{k-1}$, and $z = (z_1, z_2, \dots, z_{e_k})$ in $\mathcal{P}_k$. Then, we may evaluate

$$\begin{aligned} \langle f_{j,w}, s_k^{-1/2} x_k[n] \rangle &= \sum_{m \in \mathbb{Z}} h_k[n s_k - m] \langle f_{j,w}, M_k P_k x_{k-1}[m] \rangle \\ &= \sum_{m \in \mathbb{Z}} h_k[n s_k - m] \langle f_{j,w}, \varphi_k(P_k x_{k-1}[m]) \rangle \\ &= \sum_{m \in \mathbb{Z}} h_k[n s_k - m] f_{j,w}(P_k x_{k-1}[m]) \\ &= \sum_{m \in \mathbb{Z}} h_k[n s_k - m] \langle w, x_{k-1}[m+j] \rangle \\ &= \sum_{m \in \mathbb{Z}} h_k[n s_k + j - m] \langle w, x_{k-1}[m] \rangle \\ &= (h_k * \langle w, x_{k-1} \rangle)[n s_k + j], \end{aligned}$$

where, with an abuse of notation, $\langle w, x_{k-1}\rangle$ is the real-valued discrete signal such that $\langle w, x_{k-1}\rangle[n] = \langle w, x_{k-1}[n]\rangle$. Since integers of the form $(ns_k + j)$ cover all of $\mathbb{Z}$ according to the assumption $e_k \geq s_k$, we have a full view of the signal $(h_k * \langle w, x_{k-1}\rangle)$ on $\mathbb{Z}$. We will now follow the same reasoning as in the proof of Lemma 2.2 to recover $\langle w, x_{k-1}\rangle$:

$$\langle w, x_{k-1}\rangle = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(h_k * \langle w, x_{k-1}\rangle)}{\mathcal{F}(h_k)}\right),$$

where $\mathcal{F}$ is the Fourier transform. Since the signals involved there are discrete, their Fourier transform are periodic with period 2π, and we note that $\mathcal{F}(h_k)$ is strictly positive and bounded away from zero. The signal x_{k-1} is then recovered exactly as in the proof of Lemma 2.2 by considering for w the elements of an orthonormal basis of $\mathcal{H}_{k-1}$. □

2.C.3 Proof of Proposition 2.4

Proof. Define $(MPA)_{k:j} := M_k P_k A_{k-1} M_{k-1} P_{k-1} A_{k-2} \cdots M_j P_j A_{j-1}$. Using the fact that $\|A_k\| \leq 1$, $\|P_k\| = 1$ and M_k is non-expansive, we obtain

$$\begin{aligned}
\|\Phi_n(L_\tau x) - \Phi_n(x)\| &= \|A_n(MPA)_{n:2} M_1 P_1 A_0 L_\tau x - A_n(MPA)_{n:2} M_1 P_1 A_0 x\| \\
&\leq \|A_n(MPA)_{n:2} M_1 P_1 A_0 L_\tau x - A_n(MPA)_{n:2} M_1 L_\tau P_1 A_0 x\| \\
&\quad + \|A_n(MPA)_{n:2} M_1 L_\tau P_1 A_0 x - A_n(MPA)_{n:2} M_1 P_1 A_0 x\| \\
&\leq \|[P_1 A_0, L_\tau]\|\|x\| \\
&\quad + \|A_n(MPA)_{n:2} M_1 L_\tau P_1 A_0 x - A_n(MPA)_{n:2} M_1 P_1 A_0 x\|.
\end{aligned}$$

Note that M_1 is defined point-wise, and thus commutes with L_τ:

$$M_1 L_\tau x(u) = \varphi_1(L_\tau x(u)) = \varphi_1(x(u - \tau(u)) = M_1 x(u - \tau(u)) = L_\tau M_1 x(u).$$

By noticing that $\|M_1 P_1 A_0 x\| \leq \|x\|$, we can expand the second term above in the same way. Repeating this by induction yields

$$\begin{aligned}
\|\Phi_n(L_\tau x) - \Phi_n(x)\| &\leq \sum_{k=1}^{n} \|[P_k A_{k-1}, L_\tau]\|\|x\| + \|A_n L_\tau (MPA)_{n:1} x - A_n (MPA)_{n:1} x\| \\
&\leq \sum_{k=1}^{n} \|[P_k A_{k-1}, L_\tau]\|\|x\| + \|A_n L_\tau - A_n\|\|x\|,
\end{aligned}$$

and the result follows by decomposing $A_n L_\tau = [A_n, L_\tau] + L_\tau A_n$ and applying the triangle inequality. □

2.C.4 Proof of Lemma 2.5

Proof. The proof follows in large parts the methodology introduced by Mallat (2012) in the analysis of the stability of the scattering transform. More precisely, we will follow in part the proof of Lemma E.1 of Mallat (2012). The kernel (in the sense of Lemma 2.A.1) of A_σ is $h_\sigma(z-u) = \sigma^{-d} h(\frac{z-u}{\sigma})$. Throughout the proof, we will use the following bounds

on the decay of h for simplicity, as in Mallat (2012):[2]

$$|h(u)| \leq \frac{C_h}{(1+|u|)^{d+2}}$$
$$|\nabla h(u)| \leq \frac{C'_h}{(1+|u|)^{d+2}},$$

which are satisfied for the Gaussian function h thanks to its exponential decay.

We now decompose the commutator

$$[L_c A_\sigma, L_\tau] = L_c A_\sigma L_\tau - L_\tau L_c A_\sigma = L_c(A_\sigma - L_c^{-1} L_\tau L_c A_\sigma L_\tau^{-1}) L_\tau = L_c T L_\tau,$$

with $T := A_\sigma - L_c^{-1} L_\tau L_c A_\sigma L_\tau^{-1}$. Hence,

$$\|[L_c A_\sigma, L_\tau]\| \leq \|L_c\| \|L_\tau\| \|T\|.$$

We have $\|L_c\| = 1$ since the translation operator L_c preserves the norm. Note that we have

$$2^{-d} \leq (1 - \|\nabla\tau\|_\infty)^d \leq \det(I - \nabla\tau(u)) \leq (1 + \|\nabla\tau\|_\infty)^d \leq 2^d, \tag{2.23}$$

for all $u \in \mathbb{R}^d$. Thus, for $f \in L^2(\mathbb{R}^d)$,

$$\begin{aligned}\|L_\tau f\|^2 &= \int_{\mathbb{R}^d} |f(z - \tau(z))|^2 dz = \int_{\mathbb{R}^d} |f(u)|^2 \det(I - \nabla\tau(u))^{-1} du \\ &\leq (1 - \|\nabla\tau\|_\infty)^{-d} \|f\|^2,\end{aligned}$$

such that $\|L_\tau\| \leq (1 - \|\nabla\tau\|_\infty)^{-d/2} \leq 2^{d/2}$. This yields

$$\|[L_c A_\sigma, L_\tau]\| \leq 2^{d/2} \|T\|.$$

Kernel of T. We now show that T is an integral operator and describe its kernel. Let $\xi = (I - \tau)^{-1}$, so that $L_\tau^{-1} f(z) = f(\xi(z))$ for any function f in $L^2(\mathbb{R}^d)$. We have

$$\begin{aligned}A_\sigma L_\tau^{-1} f(z) &= \int h_\sigma(z - v) f(\xi(v)) dv \\ &= \int h_\sigma(z - u + \tau(u)) f(u) \det(I - \nabla\tau(u)), du\end{aligned}$$

using the change of variable $v = u - \tau(u)$, giving $\left|\frac{dv}{du}\right| = \det(I - \nabla\tau(u))$. Then note that $L_c^{-1} L_\tau L_c f(z) = L_\tau L_c f(z + c) = L_c f(z + c - \tau(z + c)) = f(z - \tau(z + c))$. This yields the following kernel for the operator T:

$$k(z, u) = h_\sigma(z - u) - h_\sigma(z - \tau(z + c) - u + \tau(u)) \det(I - \nabla\tau(u)). \tag{2.24}$$

A similar operator appears in Lemma E.1 of Mallat (2012), whose kernel is identical to (2.24) when $c = 0$.

Like Mallat (2012), we decompose $T = T_1 + T_2$, with kernels

$$\begin{aligned}k_1(z, u) &= h_\sigma(z - u) - h_\sigma((I - \nabla\tau(u))(z - u)) \det(I - \nabla\tau(u)) \\ k_2(z, u) &= \det(I - \nabla\tau(u)) \left(h_\sigma((I - \nabla\tau(u))(z - u)) - h_\sigma(z - \tau(z + c) - u + \tau(u))\right).\end{aligned}$$

The kernel $k_1(z, u)$ appears in (Mallat, 2012), whereas the kernel $k_2(z, u)$ involves a shift c which is not present in (Mallat, 2012). For completeness, we include the proof of the bound for both operators, even though only dealing with k_2 requires slightly new developments.

[2] Note that a more precise analysis may be obtained by using finer decay bounds.

Bound on $\|T_1\|$. We can write $k_1(z,u) = \sigma^{-d} g(u, (z-u)/\sigma)$ with

$$\begin{aligned} g(u,v) &= h(v) - h((I - \nabla\tau(u))v)\det(I - \nabla\tau(u)) \\ &= (1 - \det(I - \nabla\tau(u)))h((I - \nabla\tau(u))v) + h(v) - h((I - \nabla\tau(u))v). \end{aligned}$$

Using the fundamental theorem of calculus on h, we have

$$h(v) - h((I - \nabla\tau(u))v) = \int_0^1 \langle \nabla h((I + (t-1)\nabla\tau(u))v), \nabla\tau(u)v\rangle dt.$$

Noticing that

$$|(I + (t-1)\nabla\tau(u))v| \geq (1 - \|\nabla\tau\|_\infty)|v| \geq (1/2)|v|,$$

and that $\det(I - \nabla\tau(u))) \geq (1 - \|\nabla\tau\|_\infty)^d \geq 1 - d\|\nabla\tau\|_\infty$, we bound each term as follows

$$\begin{aligned} |(1 - \det(I - \nabla\tau(u)))h((I - \nabla\tau(u))v)| &\leq d\|\nabla\tau\|_\infty \frac{C_h}{(1 + \frac{1}{2}|v|)^{d+2}} \\ \left|\int_0^1 \langle \nabla h((I + (t-1)\nabla\tau(u))v), \nabla\tau(u)v\rangle dt\right| &\leq \|\nabla\tau\|_\infty \frac{C_h'|v|}{(1 + \frac{1}{2}|v|)^{d+2}}. \end{aligned}$$

We thus have

$$|g(u,v)| \leq \|\nabla\tau\|_\infty \frac{C_h d + C_h'|v|}{(1 + \frac{1}{2}|v|)^{d+2}}.$$

Using appropriate changes of variables in order to bound $\int |k_1(z,u)|du$ and $\int |k_1(z,u)|dz$, Schur's test yields

$$\|T_1\| \leq C_1 \|\nabla\tau\|_\infty, \tag{2.25}$$

with

$$C_1 = \int_{\mathbb{R}^d} \frac{C_h d + C_h'|v|}{(1 + \frac{1}{2}|v|)^{d+2}} dv$$

Bound on $\|T_2\|$. Let $\alpha(z,u) = \tau(z+c) - \tau(u) - \nabla\tau(u)(z-u)$, and note that we have

$$\begin{aligned} |\alpha(z,u)| &\leq |\tau(z+c) - \tau(u)| + |\nabla\tau(u)(z-u)| \\ &\leq \|\nabla\tau\|_\infty |z + c - u| + \|\nabla\tau\|_\infty |z - u| \\ &\leq \|\nabla\tau\|_\infty (|c| + 2|z-u|). \end{aligned} \tag{2.26}$$

The fundamental theorem of calculus yields

$$k_2(z,u) = -\det(I - \nabla\tau(u)) \int_0^1 \langle \nabla h_\sigma(z - \tau(z+c) - u + \tau(u) - t\alpha(z,u)), \alpha(z,u)\rangle dt.$$

We note that $|\det(I - \nabla\tau(u))| \leq 2^d$, and $\nabla h_\sigma(v) = \sigma^{-d-1}\nabla h(v/\sigma)$. Using the change of variable $z' = (z-u)/\sigma$, we obtain

$$\begin{aligned} &\int |k_2(z,u)|dz \\ &\quad \leq 2^d \int \int_0^1 \left|\nabla h\left(z' + \frac{\tau(u + \sigma z' + c) - \tau(u) - t\alpha(u + \sigma z', u)}{\sigma}\right)\right| \left|\frac{\alpha(u + \sigma z', u)}{\sigma}\right| dt dz'. \end{aligned}$$

We can use the upper bound (2.26), together with our assumption $|c| \leq \kappa\sigma$:

$$\left|\frac{\alpha(u+\sigma z', u)}{\sigma}\right| \leq \|\nabla\tau\|_\infty(\kappa + 2|z'|). \tag{2.27}$$

Separately, we have $|\nabla h(v(z'))| \leq C'_h/(1+|v(z')|)^{d+2}$, with

$$v(z') := z' + \frac{\tau(u+\sigma z'+c) - \tau(u) - t\alpha(u+\sigma z', u)}{\sigma}.$$

For $|z'| > 2\kappa$, we have

$$\begin{aligned}\left|\frac{\tau(u+\sigma z'+c) - \tau(u) - t\alpha(u+\sigma z', u)}{\sigma}\right| &= \left|t\nabla\tau(u)z' + (1-t)\frac{\tau(u+\sigma z'+c)-\tau(u)}{\sigma}\right| \\ &\leq t\|\nabla\tau\|_\infty|z'| + (1-t)\|\nabla\tau\|_\infty(|z'|+\kappa) \\ &\leq \frac{3}{2}\|\nabla\tau\|_\infty|z'| \leq \frac{3}{4}|z'|,\end{aligned}$$

and hence, using the reverse triangle inequality, $|v(z')| \geq |z'| - \frac{3}{4}|z'| = \frac{1}{4}|z'|$. This yields the upper bound

$$|\nabla h(v(z'))| \leq \begin{cases} C'_h, & \text{if } |z'| \leq 2\kappa \\ \frac{C'_h}{(1+\frac{1}{4}|z'|)^{d+2}}, & \text{if } |z'| > 2\kappa. \end{cases} \tag{2.28}$$

Combining these two bounds, we obtain

$$\int |k_2(z,u)|dz \leq C_2\|\nabla\tau\|_\infty,$$

with

$$C_2 := 2^d C'_h\left(\int_{|z'|<2\kappa}(\kappa+2|z'|)dz' + \int_{|z'|>2\kappa}\frac{\kappa+2|z'|}{(1+\frac{1}{4}|z'|)^{d+2}}dz'\right).$$

Note that the dependence of the first integral on κ is of order k^{d+1}. Following the same steps with the change of variable $u' = (z-u)/\sigma$, we obtain the bound $\int |k_2(z,u)|du \leq C_2\|\nabla\tau\|_\infty$. Schur's test then yields

$$\|T_2\| \leq C_2\|\nabla\tau\|_\infty. \tag{2.29}$$

We have thus proven

$$\|[L_c A_\sigma, L_\tau]\| \leq 2^{d/2}\|T\| \leq 2^{d/2}(C_1+C_2)\|\nabla\tau\|_\infty.$$

□

2.C.5 Discussion and Proof of Norm Preservation

We now state a result which shows that while the kernel representation may lose some of the energy of the original signal, it preserves a part of it, ensuring that the stability bound in Theorem 2.7 is non-trivial. We consider in this section the full kernel representation, including a prediction layer, which is given by $\Phi(x) = \varphi_{n+1}(\Phi_n(x))$, where φ_{n+1} is the

kernel feature map of either a Gaussian kernel (2.7) with $\alpha = 1$, or a linear kernel (2.6). In both cases, φ_{n+1} is non-expansive, which yields

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq \|\Phi_n(L_\tau x) - \Phi_n(x)\|,$$

such that the stability result of Theorem 2.7 also applies to Φ. For the Gaussian case, we trivially have a representation with norm 1, which trivially shows a preservation of norm, while for the linear case, at least part of the signal energy is preserved, in particular the energy in the low frequencies, which is predominant, for instance, in natural images (Torralba and Oliva, 2003).

Lemma 2.15 (Norm preservation)**.** *For the two choices of prediction layers, $\Phi(x)$ satisfies*

$$\|\Phi(x)\| = 1 \quad \textit{(Gaussian)}, \qquad \|\Phi(x)\| \geq \|A_n A_{n-1} \ldots A_0 x\| \quad \textit{(Linear)}.$$

It follows that the representation Φ is not contractive:

$$\sup_{x, x' \in L^2(\mathbb{R}^d, \mathcal{H}_0)} \frac{\|\Phi(x) - \Phi(x')\|}{\|x - x'\|} = 1. \tag{2.30}$$

Proof. We begin by studying $\|\Phi(x)\|$. The Gaussian case is trivial since the Gaussian kernel mapping φ_{n+1} maps all points to the sphere. In the linear case, we have

$$\begin{aligned}
\|\Phi(x)\|^2 = \|\Phi_n(x)\|^2 &= \|A_n M_n P_n x_{n-1}\|^2 \\
&= \int \|A_n M_n P_n x_{n-1}(u)\|^2 du \\
&= \int \langle \int h_{\sigma_n}(u-v) M_n P_n x_{n-1}(v) dv, \int h_{\sigma_n}(u-v') M_n P_n x_{n-1}(v') dv' \rangle du \\
&= \int\int\int h_{\sigma_n}(u-v) h_{\sigma_n}(u-v') \langle \varphi_n(P_n x_{n-1}(v)), \varphi_n(P_n x_{n-1}(v')) \rangle dv dv' du \\
&\geq \int\int\int h_{\sigma_n}(u-v) h_{\sigma_n}(u-v') \langle P_n x_{n-1}(v), P_n x_{n-1}(v') \rangle dv dv' du \\
&= \int \|A_n P_n x_{n-1}(u)\|^2 du = \|A_n P_n x_{n-1}\|^2,
\end{aligned}$$

where the inequality follows from $\langle \varphi_n(z), \varphi_n(z') \rangle = K_n(z, z') \geq \langle z, z' \rangle$ (see Lemma 2.1). Using Fubini's theorem and the fact that A_n commutes with translations, we have

$$\begin{aligned}
\|A_n P_n x_{n-1}\|^2 &= \int_{S_n} \|A_n L_v x_{n-1}\|^2 d\nu_n(v) \\
&= \int_{S_n} \|L_v A_n x_{k-1}\|^2 d\nu_n(v) \\
&= \int_{S_n} \|A_n x_{k-1}\|^2 d\nu_n(v) \\
&= \|A_n x_{n-1}\|^2,
\end{aligned}$$

where we used the fact that translations L_v preserve the norm. Note that we have

$$A_n x_{n-1} = A_n A_{n-1} M_{n-1} P_{n-1} x_{n-2} = A_{n,n-1} M_{n-1} P_{n-1} x_{n-2},$$

where $A_{n,n-1}$ is an integral operator with positive kernel $h_{\sigma_n} * h_{\sigma_{n-1}}$. Repeating the above relation then yields

$$\|\Phi(x)\|^2 \geq \|A_n x_{n-1}\|^2 \geq \|A_n A_{n-1} x_{n-1}\|^2 \geq \ldots \geq \|A_n A_{n-1} \ldots A_0 x\|^2,$$

and the result follows.

We now show (2.30). By our assumptions on φ_{n+1} and on the operators A_k, M_k, P_k, we have that Φ is non-expansive, so that

$$\sup_{x,x' \in L^2(\mathbb{R}^d, \mathcal{H}_0)} \frac{\|\Phi(x) - \Phi(x')\|}{\|x - x'\|} \leq 1.$$

It then suffices to show that one can find x, x' such that the norm ratio $\frac{\|\Phi(x)-\Phi(x')\|}{\|x-x'\|}$ is arbitrarily close to 1. In particular, we begin by showing that for any signal $x \neq 0$ we have

$$\lim_{\lambda \to 1} \frac{\|\Phi(\lambda x) - \Phi(x)\|}{\|\lambda x - x\|} \geq \frac{\|A_\sigma x\|}{\|x\|}, \tag{2.31}$$

where A_σ is the pooling operator with scale $\sigma = (\sigma_n^2 + \sigma_{n-1}^2 + \ldots + \sigma_1^2)^{1/2}$, and the result will follow by considering appropriate signals x that make this lower bound arbitrarily close to 1.

Note that by homogeneity of the kernels maps φ_k (which follows from the homogeneity of kernels K_k), and by linearity of the operators A_k and P_k, we have $\Phi_n(\lambda x) = \lambda \Phi_n(x)$ for any $\lambda \geq 0$. Taking $\lambda > 0$, we have

$$\|\Phi_n(\lambda x) - \Phi_n(x)\| = (\lambda - 1)\|\Phi_n(x)\| \geq (\lambda - 1)\|A_n A_{n-1} \ldots A_0 x\| = (\lambda - 1)\|A_\sigma x\|,$$

adapting Lemma 2.15 to the representation Φ_n. Thus,

$$\lim_{\lambda \to 1} \frac{\|\Phi_n(\lambda x) - \Phi_n(x)\|}{\|\lambda x - x\|} \geq \frac{\|A_\sigma x\|}{\|x\|}.$$

When φ_{n+1} is linear, we immediately obtain (2.31) since $\|\Phi(\lambda x) - \Phi(x)\| = \|\Phi_n(\lambda x) - \Phi_n(x)\|$. For the Gaussian case, we have

$$\begin{aligned}
\|\Phi(\lambda x) - \Phi(x)\|^2 &= 2 - 2e^{-\frac{1}{2}\|\Phi_n(\lambda x) - \Phi_n(x)\|^2} \\
&= 2 - 2e^{-\frac{1}{2}(\lambda-1)^2\|\Phi_n(x)\|^2} \\
&= (\lambda - 1)^2 \|\Phi_n(x)\|^2 + o((\lambda - 1)^2) \\
&= \|\Phi_n(\lambda x) - \Phi_n(x)\|^2 + o((\lambda - 1)^2),
\end{aligned}$$

which yields (2.31).

By considering a Gaussian signal with scale $\tau \gg \sigma$, we can make $\frac{\|A_\sigma x\|}{\|x\|}$ arbitrarily close to 1 by taking an arbitrarily large τ. It follows that

$$\sup_x \lim_{\lambda \to 1} \frac{\|\Phi(\lambda x) - \Phi(x)\|}{\|\lambda x - x\|} = 1,$$

which yields the result. □

2.C.6 Proof of Lemma 2.9

Proof. We have

$$\begin{aligned}
Px((u,\eta)) &= (v \in \tilde{S} \mapsto x((u,\eta)(v,0))) \\
&= (v \in \tilde{S} \mapsto x((u + R_\eta v, \eta))) \\
&= (v \in R_\eta \tilde{S} \mapsto x((u+v,\eta))) \\
Ax((u,\eta)) &= \int_{\mathbb{R}^2} x((u,\eta)(v,0))h(v)dv \\
&= \int_{\mathbb{R}^2} x((u + R_\eta v, \eta))h(v)dv \\
&= \int_{\mathbb{R}^2} x((v,\eta))h(R_{-\eta}(v-u))dv \\
&= \int_{\mathbb{R}^2} x((v,\eta))h(u-v)dv,
\end{aligned}$$

where the last equality uses the circular symmetry of a Gaussian around the origin. For a diffeomorphism τ, we denote by L_τ the action operator given by $L_\tau x((u,\eta)) = x((\tau(u),0)^{-1}(u,\eta)) = x((u-\tau(u),\eta))$. If we denote $x(\cdot,\eta)$ the $L^2(\mathbb{R}^2)$ signal obtained from a signal $x \in L^2(G)$ at a fixed angle, we have shown

$$\begin{aligned}
(Px)(\cdot,\eta) &= \tilde{P}_\eta(x(\cdot,\eta)) \\
(Ax)(\cdot,\eta) &= \tilde{A}(x(\cdot,\eta)) \\
(L_\tau x)(\cdot,\eta) &= \tilde{L}_\tau(x(\cdot,\eta)),
\end{aligned}$$

where $\tilde{P}_\eta, \tilde{A}, \tilde{L}_\tau$ are defined on $L^2(\mathbb{R}^2)$ as in Section 2.2, with a rotated patch $R_\eta\tilde{S}$ for $\tilde{P}_\eta$. Then, we have, for a signal $x \in L^2(G)$,

$$\begin{aligned}
\|[PA, L_\tau]x\|^2_{L^2(G)} &= \int \|([PA, L_\tau]x)(\cdot,\eta)\|^2_{L^2(\mathbb{R}^2)} d\mu_c(\eta) \\
&= \int \|[\tilde{P}_\eta\tilde{A}, \tilde{L}_\tau](x(\cdot,\eta))\|^2_{L^2(\mathbb{R}^2)} d\mu_c(\eta) \\
&\leq \int \|[\tilde{P}_\eta\tilde{A}, \tilde{L}_\tau]\|^2 \|x(\cdot,\eta)\|^2_{L^2(\mathbb{R}^2)} d\mu_c(\eta) \\
&\leq \left(\sup_\eta \|[\tilde{P}_\eta\tilde{A}, \tilde{L}_\tau]\|^2\right) \|x\|^2_{L^2(G)},
\end{aligned}$$

so that $\|[PA, L_\tau]\|_{L^2(G)} \leq \sup_\eta \|[\tilde{P}_\eta\tilde{A}, \tilde{L}_\tau]\|^2_{L^2(\mathbb{R}^2)}$. Note that we have $\sup_{c \in R_\eta\tilde{S}} |c| = \sup_{c\in\tilde{S}} |c| \leq \kappa\sigma$, since rotations preserve the norm, so that we can bound each $\|[\tilde{P}_\eta\tilde{A}, \tilde{L}_\tau]\|$ as in Section 2.3.1 to obtain the desired result. Similarly, $\|L_\tau A - A\|$ can be bounded as in Section 2.3.1.

□

2.C.7 Proof of Theorem 2.10

Proof. First, note that A_c can be written as an integral operator

$$A_c x(u) = \int x((v,\eta))k(u,(v,\eta))d\mu((v,\eta)),$$

with $k(u,(v,\eta)) = \delta_u(v)$, where δ denotes the Dirac delta function. We have

$$\int |k(u,(v,\eta))| d\mu((v,\eta)) = \int |k(u,(v,\eta))| du = 1.$$

By Schur's test, we thus obtain $\|A_c\| \leq 1$. Then, note that $(\tau(u),\theta) = (0,\theta)(R_{-\theta}\tau(u),0)$, so that $L_{(\tau,\theta)} = L_{(0,\theta)}L_{\tau_\theta}$, where we write $\tau_\theta(u) = R_{-\theta}\tau(u)$. Additionally, it is easy to see that $A_c L_{(0,\theta)} = A_c$. We have

$$\begin{aligned}
\|A_c\Phi_n(L_{(\tau,\theta)}x) - A_c\Phi_n(x)\| &= \|A_c\Phi_n(L_{(0,\theta)}L_{\tau_\theta}x) - A_c\Phi_n(x)\| \\
&= \|A_c L_{(0,\theta)}\Phi_n(L_{\tau_\theta}x) - A_c\Phi_n(x)\| \\
&= \|A_c\Phi_n(L_{\tau_\theta}x) - A_c\Phi_n(x)\| \\
&\leq \|\Phi_n(L_{\tau_\theta}x) - \Phi_n(x)\|,
\end{aligned}$$

using the fact that the representation Φ_n is equivariant to roto-translations by construction.

We conclude by using Lemma 2.9 together with an adapted version of Proposition 2.4, and by noticing that $\|\nabla\tau_\theta\|_\infty = \|\nabla\tau\|_\infty$ and $\|\tau_\theta\|_\infty = \|\tau\|_\infty$. □

2.D Proofs Related to the Construction of CNNs in the RKHS

2.D.1 Proof of Lemma 2.11

Proof. Here, we drop all indices k since there is no ambiguity. We will now characterize the functional space $\mathcal{H}$ by following the same strategy as Zhang et al. (2016, 2017b) for the non-homogeneous Gaussian and inverse polynomial kernels on Euclidean spaces. Using the Maclaurin expansion of κ, we can define the following explicit feature map for the dot-product kernel $K_{\text{dp}}(z,z') := \kappa(\langle z,z'\rangle)$, for any z in the unit-ball of $\mathcal{P}$:

$$\begin{aligned}
\psi_{\text{dp}}(z) &= \left(\sqrt{b_0}, \sqrt{b_1}z, \sqrt{b_2}z\otimes z, \sqrt{b_3}z\otimes z\otimes z, \ldots\right) \\
&= \left(\sqrt{b_j}z^{\otimes j}\right)_{j\in\mathbb{N}},
\end{aligned} \tag{2.32}$$

where $z^{\otimes j}$ denotes the tensor product of order j of the vector z. Technically, the explicit mapping lives in the Hilbert space $\oplus_{j=0}^n \otimes^j \mathcal{P}$, where $\oplus$ denotes the direct sum of Hilbert spaces, and with the abuse of notation that $\otimes^0\mathcal{P}$ is simply $\mathbb{R}$. Then, we have that $K_{\text{dp}}(z,z') = \langle\psi(z),\psi(z')\rangle$ for all z,z' in the unit ball of $\mathcal{P}$. Similarly, we can construct an explicit feature map for the homogeneous dot-product kernels (2.2):

$$\begin{aligned}
\psi_{\text{hdp}}(z) &= \left(\sqrt{b_0}\|z\|, \sqrt{b_1}z, \sqrt{b_2}\|z\|^{-1}z\otimes z, \sqrt{b_3}\|z\|^{-2}z\otimes z\otimes z, \ldots\right) \\
&= \left(\sqrt{b_j}\|z\|^{1-j}z^{\otimes j}\right)_{j\in\mathbb{N}}.
\end{aligned} \tag{2.33}$$

From these mappings, we may now conclude the proof by following the same strategy as Zhang et al. (2016, 2017b). By first considering the restriction of K to unit-norm vectors z,

$$\sigma(\langle w,z\rangle) = \sum_{j=0}^{+\infty} a_j\langle w,z\rangle^j = \sum_{j=0}^{+\infty} a_j\langle w^{\otimes j}, z^{\otimes j}\rangle = \langle\bar{w},\psi(z)\rangle,$$

where

$$\bar{w} = \left(\frac{a_j}{\sqrt{b_j}} w^{\otimes j} \right)_{j \in \mathbb{N}}.$$

Then, the norm of $\bar{w}$ is

$$\|\bar{w}\|^2 = \sum_{j=0}^{+\infty} \frac{a_j^2}{b_j} \|w^{\otimes j}\|^2 = \sum_{j=0}^{+\infty} \frac{a_j^2}{b_j} \|w\|^{2j} = C_\sigma^2(\|w\|^2) < +\infty.$$

Using Theorem 2.A.1, we conclude that f is in the RKHS of K, with norm $\|f\| \leq C_\sigma(\|w\|^2)$. Finally, we extend the result to non unit-norm vectors z with similar calculations and we obtain the desired result. □

2.D.2 CNN construction and RKHS norm

In this section, we describe the space of functions (RKHS) $\mathcal{H}_{\mathcal{K}_n}$ associated to the kernel $\mathcal{K}_n(x_0, x_0') = \langle x_n, x_n' \rangle$ defined in (2.6), where x_n, x_n' are the final representations given by Eq. (2.5), in particular showing it contains the set of CNNs with activations described in Section 2.4.1.

Construction of a CNN in the RKHS.

Let us consider the definition of the CNN presented in Section 2.4. We will show that it can be seen as a point in the RKHS of $\mathcal{K}_n$. According to Lemma 2.11, we consider $\mathcal{H}_k$ that contains all functions of the form $z \in \mathcal{P}_k \mapsto \|z\|\sigma(\langle w, z \rangle / \|z\|)$, with $w \in \mathcal{P}_k$.

We recall the intermediate quantities introduced in Section 2.4. That is, we define the initial quantities $f_1^i \in \mathcal{H}_1, g_1^i \in \mathcal{P}_1$ for $i = 1, \ldots, p_1$ such that

$$\begin{aligned} g_1^i = w_1^i \in L^2(S_1, \mathbb{R}^{p_0}) = L^2(S_1, \mathcal{H}_0) = \mathcal{P}_1 \\ f_1^i(z) = \|z\|\sigma(\langle g_i^0, z \rangle / \|z\|) \quad \text{for } z \in \mathcal{P}_1, \end{aligned}$$

and we recursively define, from layer k–1, the quantities $f_k^i \in \mathcal{H}_k, g_k^i \in \mathcal{P}_k$ for $i = 1, \ldots, p_k$:

$$\begin{aligned} g_k^i(v) &= \sum_{j=1}^{p_{k-1}} w_k^{ij}(v) f_{k-1}^j \quad \text{where} \quad w_k^i(v) = (w_k^{ij}(v))_{j=1,\ldots,p_{k-1}} \\ f_k^i(z) &= \|z\|\sigma(\langle g_k^i, z \rangle / \|z\|) \quad \text{for } z \in \mathcal{P}_k. \end{aligned}$$

Then, we will show that $\tilde{z}_k^i(u) = f_k^i(P_k x_{k-1}(u)) = \langle f_k^i, M_k P_k x_{k-1}(u) \rangle$, which correspond to feature maps at layer k and index i in a CNN. Indeed, this is easy to see

for $k = 1$ by construction with filters $w_1^i(v)$, and for $k \geq 2$, we have

$$\begin{aligned}
\tilde{z}_k^i(u) &= n_k(u)\sigma\Big(\langle w_k^i, P_k z_{k-1}(u)\rangle / n_k(u)\Big) \\
&= n_k(u)\sigma\Big(\langle w_k^i, P_k A_{k-1}\tilde{z}_{k-1}(u)\rangle / n_k(u)\Big) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\sum_{j=1}^{p_{k-1}}\int_{S_k} w_k^{ij}(v) A_{k-1}\tilde{z}_{k-1}^j(u+v)d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\sum_{j=1}^{p_{k-1}}\int_{S_k} w_k^{ij}(v)\langle f_{k-1}^j, A_{k-1}M_{k-1}P_{k-1}x_{k-2}(u+v)\rangle d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\int_{S_k}\langle g_k^i(v), A_{k-1}M_{k-1}P_{k-1}x_{k-2}(u+v)\rangle d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\int_{S_k}\langle g_k^i(v), x_{k-1}(u+v)\rangle d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\langle g_k^i(v), P_k x_{k-1}(u)\rangle\right) \\
&= f_k^i(P_k x_{k-1}(u)),
\end{aligned}$$

where $n_k(u) := \|P_k x_{k-1}(u)\|$. Note that we have used many times the fact that A_k operates on each channel independently when applied to a finite-dimensional map.

The final prediction function is of the form $f_\sigma(x_0) = \langle w_{n+1}, z_n\rangle$ with w_{n+1} in $L^2(\mathbb{R}^d, \mathbb{R}^{p_n})$. Then, we can define the following function g_σ in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ such that

$$g_\sigma(u) = \sum_{j=1}^{p_n} w_{n+1}^j(u) f_n^j,$$

which yields

$$\begin{aligned}
\langle g_\sigma, x_n\rangle &= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u)\langle f_n^j, x_n(u)\rangle du \\
&= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u)\langle f_n^j, A_n M_n P_n x_{n-1}(u)\rangle du \\
&= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u) A_n \tilde{z}_n^j(u) du \\
&= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u) z_n^j(u) du \\
&= \sum_{j=1}^{p_n}\langle w_{n+1}^j, z_n^j\rangle = f_\sigma(x_0),
\end{aligned}$$

which corresponds to a linear layer after pooling. Since the RKHS of $\mathcal{K}_n$ in the linear case (2.6) contains all functions of the form $f(x_0) = \langle g, x_n\rangle$, for g in $L^2(\mathbb{R}^d, \mathcal{H}_n)$, we have that f_σ is in the RKHS.

Proof of Proposition 2.13

Proof. As shown in Lemma 2.11, the norm of a function $f : z \in \mathcal{P}_k \mapsto \|z\|\sigma(\langle w, z\rangle/\|z\|)$ in $\mathcal{H}_k$ is bounded by $C_\sigma(\|w\|^2)$, where C_σ depends on the activation σ. We then have

$$\begin{aligned}
\|f_1^i\|^2 &\leq C_\sigma^2(\|w_1^i\|_2^2) \quad \text{where} \quad \|w_1^i\|_2^2 = \int_{S_1} \|w_1^i(v)\|^2 d\nu_1(v) \\
\|f_k^i\|^2 &\leq C_\sigma^2(\|g_k^i\|^2) \\
\|g_k^i\|^2 &= \int_{S_k} \|\sum_{j=1}^{p_{k-1}} w_k^{ij}(v) f_{k-1}^j\|^2 d\nu_k(v) \\
&\leq p_{k-1} \sum_{j=1}^{p_{k-1}} \left(\int_{S_k} |w_k^{ij}(v)|^2 d\nu_k(v) \right) \|f_{k-1}^j\|^2 \\
&= p_{k-1} \sum_{j=1}^{p_{k-1}} \|w_k^{ij}\|_2^2 \|f_{k-1}^j\|^2,
\end{aligned}$$

where in the last inequality we use $\|a_1 + \ldots + a_n\|^2 \leq n(\|a_1\|^2 + \ldots + \|a_n\|^2)$. Since C_σ^2 is monotonically increasing (typically exponentially in its argument), we have for $k = 1, \ldots, n-1$ the recursive relation

$$\|f_k^i\|^2 \leq C_\sigma^2 \left(p_{k-1} \sum_{j=1}^{p_{k-1}} \|w_k^{ij}\|_2^2 \|f_{k-1}^j\|^2 \right).$$

The norm of the final prediction function $f \in L^2(\mathbb{R}^d, \mathcal{H}_n)$ is bounded as follows, using similar arguments as well as Theorem 2.A.1:

$$\|f_\sigma\|^2 \leq \|g_\sigma\|^2 \leq p_n \sum_{j=1}^{p_n} \left(\int_{\mathbb{R}^d} |w_{n+1}^j(u)|^2 du \right) \|f_n^j\|^2.$$

This yields the desired result. □

Proof of Proposition 2.14

Proof. Define

$$\begin{aligned}
F_k &= (f_k^1, \ldots, f_k^{p_k}) \in \mathcal{H}_k^{p_k} \\
G_k &= (g_k^1, \ldots, g_k^{p_k}) \in \mathcal{P}_k^{p_k} \\
W_k(u) &= (w_k^{ij}(u))_{ij} \in \mathbb{R}^{p_k \times p_{k-1}} \quad \text{for } u \in S_k.
\end{aligned}$$

We will write, by abuse of notation, $G_k(u) = (g_k^1(u), \ldots, g_k^{p_k}(u))$ for $u \in S_k$, so that we can write $G_k(u) = W_k(u)F_{k-1}$. In particular, we have $\|G_k(u)\| \leq \|W_k(u)\|_2 \|F_{k-1}\|$. This can be seen by considering an orthonormal basis B of $\mathcal{H}_k$, and defining real-valued vectors $F_k^w = (\langle w, f_k^1\rangle, \ldots, \langle w, f_k^{p_k}\rangle)$, $G_k^w(u) = (\langle w, g_k^1(u)\rangle, \ldots, \langle w, g_k^{p_k}(u)\rangle)$ for $w \in B$. Indeed, we have $G_k^w(u) = W_k(u)F_{k-1}^w$ and hence $\|G_k^w(u)\| \leq \|W_k(u)\|_2 \|F_{k-1}^w\|$ for all $w \in B$, and we conclude using

$$\|G_k(u)\|^2 = \sum_{w \in B} \|G_k^w(u)\|^2 \leq \|W_k(u)\|_2^2 \sum_{w \in B} \|F_{k-1}^w\|^2 = \|W_k(u)\|_2^2 \, \|F_{k-1}\|^2.$$

Then, we have

$$\|G_k\|^2 = \sum_i \|g_k^i\|^2 = \sum_i \int_{S_k} \|g_k^i(u)\|^2 d\nu_k(u) = \int_{S_k} \|G_k(u)\|^2 d\nu_k(u)$$
$$\leq \int_{S_k} \|W_k(u)\|_2^2 \, \|F_{k-1}\|^2 \nu_k(u) = \|W_k\|_2^2 \, \|F_{k-1}\|^2.$$

Separately, we notice that C_σ^2 is super-additive, *i.e.*,

$$C_\sigma^2(\lambda_1^2 + \ldots + \lambda_n^2) \geq C_\sigma^2(\lambda_1^2) + \ldots + C_\sigma^2(\lambda_n^2).$$

Indeed, this follows from the definition of C_σ^2, noting that polynomials with non-negative coefficients are super-additive on non-negative numbers. Thus, we have

$$\|F_1\|^2 = \sum_{i=1}^{p_1} \|f_1^i\|^2 \leq \sum_{i=1}^{p_1} C_\sigma^2(\|w_1^i\|^2) \leq C_\sigma^2(\|W_1\|_F^2)$$
$$\|F_k\|^2 \leq \sum_{i=1}^{p_k} C_\sigma^2(\|g_k^i\|^2) \leq C_\sigma^2(\|G_k\|^2), \quad \text{for } k = 2, \ldots, n.$$

Finally, note that

$$\|g_\sigma(u)\|^2 \leq \left(\sum_{j=1}^{p_n} |w_{n+1}^j(u)| \|f_n^j\| \right)^2 \leq \|w_{n+1}(u)\|^2 \|F_n\|^2,$$

by using Cauchy-Schwarz, so that $\|g_\sigma\|^2 \leq \|w_{n+1}\|^2 \|F_n\|^2$. Thus, combining the previous relations yields

$$\|f_\sigma\|^2 \leq \|g_\sigma\|^2 \leq \|w_{n+1}\|^2 \, C_\sigma^2(\|W_n\|_2^2 \, C_\sigma^2(\|W_{n-1}\|_2^2 \ldots C_\sigma^2(\|W_1\|_F^2) \ldots)),$$

which is the desired result. □

Chapter 3

A Kernel Perspective on Regularization and Robustness of Deep Neural Networks

We propose a new point of view for regularizing deep neural networks by using the norm of a reproducing kernel Hilbert space (RKHS). Even though this norm cannot be computed, it admits upper and lower approximations leading to various practical strategies. Specifically, this perspective (i) provides a common umbrella for many existing regularization principles, including spectral norm and gradient penalties, or adversarial training, (ii) leads to new effective regularization penalties, and (iii) suggests hybrid strategies combining lower and upper bounds to get better approximations of the RKHS norm. We experimentally show this approach to be effective when learning on small datasets, or to obtain adversarially robust models. In particular, some of our approaches lead to state-of-the-art performance for robustness to ℓ_2 perturbations on the CIFAR10 dataset.

This chapter is based on the following paper:

A. Bietti, G. Mialon, D. Chen, and J. Mairal. A kernel perspective for regularizing deep neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019

The thesis author and G. Mialon are joint first authors on the paper. The thesis author focused on lower bound approaches and theoretical insights based on kernel methods, while G. Mialon focused on the upper bound approaches based on spectral norms. D. Chen worked on the applications to protein homology detection.

3.1 Introduction

Learning predictive models for complex tasks often requires large amounts of annotated data. For instance, convolutional neural networks are huge-dimensional and typically involve more parameters than training samples, which raises several challenges: achieving good generalization with small datasets is indeed difficult, which limits the deployment of such deep models to many tasks where labeled data is scarce, *e.g.*, in biology (Ching et al., 2018). Besides, imperceptible adversarial perturbations can significantly degrade the prediction quality (Szegedy et al., 2014; Biggio and Roli, 2018). These issues raise the

question of regularization as an essential tool to control the complexity of deep models, as well as their stability to small variations of their inputs.

In this chapter, we present a new perspective on regularization of deep networks, by viewing convolutional neural networks (CNNs) as elements of a RKHS following the work of Bietti and Mairal (2019a) on deep convolutional kernels. For such kernels, the RKHS contains indeed deep convolutional networks similar to generic ones—up to smooth approximations of rectified linear units. Such a point of view provides a natural regularization function, the RKHS norm, which allows us to control the variations of the predictive model and to limit its complexity for better generalization. Besides, the norm also acts as a Lipschitz constant, which provides a direct control on the stability to adversarial perturbations.

In contrast to traditional kernel methods, the RKHS norm cannot be explicitly computed in our setup. Yet, this norm admits numerous approximations—lower bounds and upper bounds—which lead to many strategies for regularization based on penalties, constraints, or combinations thereof. Depending on the chosen approximation, we recover then many existing principles such as spectral norm regularization (Cisse et al., 2017; Yoshida and Miyato, 2017; Miyato et al., 2018a; Sedghi et al., 2019), gradient penalties and double backpropagation (Drucker and Le Cun, 1991; Simon-Gabriel et al., 2019; Gulrajani et al., 2017; Roth et al., 2017, 2018; Arbel et al., 2018), adversarial training (Madry et al., 2018), and we also draw links with tangent propagation (Simard et al., 1998). For all these principles, we provide a unified viewpoint and theoretical insights, and we also introduce new variants, which we show are effective in practice when learning with few labeled data, or in the presence of adversarial perturbations.

Moreover, regularization and robustness are tightly linked in our kernel framework. Specifically, some lower bounds on the RKHS norm lead to robust optimization objectives with worst-case ℓ_2 perturbations; further, we can extend margin-based generalization bounds in the spirit of Bartlett et al. (2017); Boucheron et al. (2005) to the setting of *adversarially robust* generalization (see Schmidt et al., 2018), where an adversary can perturb test data. We also discuss connections between recent regularization strategies for training generative adversarial networks and approaches to generative modeling based on kernel two-sample tests (MMD) (Dziugaite et al., 2015; Li et al., 2017; Bińkowski et al., 2018).

Summary of the contributions.

- We introduce an RKHS perspective for regularizing deep neural networks models which provides a unified view on various practical regularization principles, together with theoretical insight and guarantees;
- By considering lower bounds to the RKHS norm, we obtain new penalties based on adversarial perturbations, adversarial deformations, or gradient norms of prediction functions, which we show to be effective in practice;
- Our RKHS point of view suggests combined strategies based on both upper and lower bounds, which we show often perform empirically best in the context of generalization from small image and biological datasets, by providing a tighter control of the RKHS norm.

Related work. The construction of hierarchical kernels and the study of neural networks in the corresponding RKHS was studied by Mairal (2016); Zhang et al. (2016, 2017b); Bietti and Mairal (2019a). Some of the regularization strategies we obtain from our kernel perspective are variants of previous approaches to adversarial robustness (Cisse et al., 2017; Madry et al., 2018; Simon-Gabriel et al., 2019; Roth et al., 2018), to improving generalization (Drucker and Le Cun, 1991; Miyato et al., 2018b; Sedghi et al., 2019; Simard et al., 1998; Yoshida and Miyato, 2017), and stable training of generative adversarial networks (Roth et al., 2017; Gulrajani et al., 2017; Arbel et al., 2018; Miyato et al., 2018a). The link between robust optimization and regularization was studied by Xu et al. (2009a,b), focusing mainly on linear models with quadratic or hinge losses. The notion of adversarial generalization was considered by Schmidt et al. (2018), who provide lower bounds on a particular data distribution. Sinha et al. (2018) provide generalization guarantees in the different setting of distributional robustness; compared to our bound, they consider expected loss instead of classification error, and their bounds do not highlight the dependence on the model complexity.

3.2 Regularization of Deep Neural Networks

In this section, we recall the kernel perspective on deep networks introduced in Chapter 2, and present upper and lower bounds on the RKHS norm of a given model, leading to various regularization strategies. For simplicity, we first consider real-valued networks and binary classification, before discussing multi-class extensions.

3.2.1 Relation between deep networks and RKHSs

Kernel methods consist of mapping data living in a set $\mathcal{X}$ to a RKHS $\mathcal{H}$ associated to a positive definite kernel K through a mapping function $\Phi : \mathcal{X} \to \mathcal{H}$, and then learning simple machine learning models in $\mathcal{H}$. Specifically, when considering a real-valued regression or binary classification problem, classical kernel methods find a prediction function $f : \mathcal{X} \to \mathbb{R}$ living in the RKHS which can be written in linear form, i.e., such that $f(x) = \langle f, \Phi(x) \rangle_{\mathcal{H}}$ for all x in $\mathcal{X}$. While explicit mapping to a possibly infinite-dimensional space is of course only an abstract mathematical operation, learning f can be done implicitly by computing kernel evaluations and typically by using convex programming (Schölkopf and Smola, 2001).

Moreover, the RKHS norm $\|f\|_{\mathcal{H}}$ acts as a natural regularization function, which controls the variations of model predictions according to the geometry induced by Φ:

$$|f(x) - f(x')| \leq \|f\|_{\mathcal{H}} \cdot \|\Phi(x) - \Phi(x')\|_{\mathcal{H}}. \tag{3.1}$$

Unfortunately, our setup does not allow us to use the RKHS norm in a traditional way since evaluating the kernel is intractable. Instead, we propose a different approach that considers explicit parameterized representations of functions contained in the RKHS, given by generic CNNs, and leverage properties of the RKHS and the kernel mapping in order to regularize when learning the network parameters.

Consider indeed a real-valued deep convolutional network $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is simply $\mathbb{R}^d$, with rectified linear unit (ReLU) activations and no bias units. By constructing an appropriate multi-layer hierarchical kernel, Bietti and Mairal (2019a) show that

the corresponding RKHS $\mathcal{H}$ contains a CNN with the same architecture and parameters as f, but with activations that are smooth approximations of ReLU. Although the model predictions might not be strictly equal, we will abuse notation and denote this approximation with smooth ReLU by f as well, with the hope that the regularization procedures derived from the RKHS model will be effective in practice on the original CNN f.

Besides, the mapping $\Phi(\cdot)$ is shown to be non-expansive:

$$\|\Phi(x) - \Phi(x')\|_{\mathcal{H}} \leq \|x - x'\|_2, \tag{3.2}$$

so that controlling $\|f\|_{\mathcal{H}}$ provides some robustness to additive ℓ_2-perturbations, by (3.1). Additionally, with appropriate pooling operations, Bietti and Mairal (2019a) show that the kernel mapping is also stable to deformations, meaning that the RKHS norm also controls robustness to translations and other transformations including scaling and rotations, which can be seen as deformations when they are small.

In contrast to standard kernel methods, where the RKHS norm is typically available in closed form, this norm is difficult to compute in our setup, and requires approximations. The following sections present upper and lower bounds on $\|f\|_{\mathcal{H}}$, with linear convolutional operations denoted by W_k for $k = 1, \ldots, L$, where L is the number of layers. Defining $\theta := \{W_k : k = 1, \ldots, L\}$, we then leverage these bounds to approximately solve the following penalized or constrained optimization problems on a training set $(x_i, y_i), i = 1, \ldots, n$:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i)) + \lambda \|f_\theta\|_{\mathcal{H}}^2 \quad \text{or} \tag{3.3}$$

$$\min_{\theta : \|f_\theta\|_{\mathcal{H}} \leq C} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i)). \tag{3.4}$$

We also note that while the construction of Bietti and Mairal (2019a) considers VGG-like networks (Simonyan and Zisserman, 2014), the regularization algorithms we obtain in practice can be easily adapted to different architectures such as residual networks (He et al., 2016).

3.2.2 Exploiting lower bounds of the RKHS norm

In this section, we devise regularization algorithms by leveraging lower bounds on $\|f\|_{\mathcal{H}}$, obtained by relying on the following variational characterization of Hilbert norms:

$$\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle_{\mathcal{H}}.$$

At first sight, this definition is not useful since the set $U = \{u \in \mathcal{H} : \|u\|_{\mathcal{H}} \leq 1\}$ may be infinite-dimensional and the inner products $\langle f, u \rangle_{\mathcal{H}}$ cannot be computed in general. Thus, we devise tractable lower bound approximations by considering smaller sets $\bar{U}$ such that $\bar{U} \subset U$.

Adversarial perturbation penalty. Thanks to the non-expansiveness of Φ, we can consider the subset $\bar{U} \subset U$ defined as $\bar{U} = \{\Phi(x+\delta) - \Phi(x) : x \in \mathcal{X}, \|\delta\|_2 \leq 1\}$, leading to the bound

$$\|f\|_{\mathcal{H}} \geq \|f\|_{\delta}^2 := \sup_{x \in \mathcal{X}, \|\delta\|_2 \leq 1} f(x+\delta) - f(x), \tag{3.5}$$

which is reminiscent of adversarial perturbations. Adding a regularization parameter $\epsilon > 0$ in front of the norm then corresponds to different sizes of perturbations:

$$\epsilon\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq \epsilon} \langle f, u \rangle_{\mathcal{H}} \geq \sup_{x \in \mathcal{X}, \|\delta\|_2 \leq \epsilon} f(x+\delta) - f(x). \tag{3.6}$$

Using this lower bound or its square as a penalty in the objective (3.3) when training a CNN provides a way to regularize. Optimizing over adversarial perturbations has been useful to obtain robust models (*e.g.*, the PGD method of Madry et al., 2018); yet our approach differs in two important ways:

(i) it involves a penalty that is decoupled from the loss term such that in principle, our penalty could be used beyond the supervised empirical risk paradigm. In contrast, PGD optimizes the robust formulation (3.7) below, which fits training data while considering perturbations on the loss.

(ii) our penalty involves a global maximization problem on the input space $\mathcal{X}$, as opposed to only maximizing on perturbations near training data. In practice, optimizing over $\mathcal{X}$ is however difficult and instead, we replace $\mathcal{X}$ by random mini-batches of examples, yielding further lower bounds on the RKHS norm. These examples may be labeled or not, in contrast to PGD that perturb labeled examples only. When using such a mini-batch, a gradient of the penalty can be obtained by first finding maximizers $\hat{x}, \hat{\delta}$ (where $\hat{x}$ is an element of the mini-batch and $\hat{\delta}$ is a perturbation), and then computing gradients of $f_\theta(\hat{x} + \hat{\delta}) - f_\theta(\hat{x})$ with respect to θ by using back-propagation. In practice, we compute the perturbations δ for each example x by using a few steps of projected gradient ascent with constant step-lengths.

Robust optimization yields another lower bound. In some contexts, our penalized approach is related to solving the robust optimization problem

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \sup_{\|\delta\|_2 \leq \epsilon} \ell(y_i, f_\theta(x_i + \delta)), \tag{3.7}$$

which is commonly considered for training adversarially robust classifiers (Wong and Kolter, 2018; Madry et al., 2018; Raghunathan et al., 2018). In particular, Xu et al. (2009b) show that the penalized and robust objectives are equivalent in the case of the hinge loss with linear predictors, when the data is non-separable. They also show the equivalence for kernel methods when considering the (intractable) full perturbation set U around each point in the RKHS $\Phi(x_i)$, that is, predictions $\langle f, \Phi(x_i) + u \rangle_{\mathcal{H}}$ with u in U. Intuitively, when a training example (x_i, y_i) is misclassified, we are in the "linear" part of the hinge loss, such that

$$\sup_{\|u\|_{\mathcal{H}} \leq \epsilon} \ell(y_i, \langle f, \Phi(x_i) + u \rangle_{\mathcal{H}}) = \ell(y_i, f(x_i)) + \epsilon\|f\|_{\mathcal{H}}.$$

For other losses such as the logistic loss, a regularization effect is still present even for correctly classified examples, though it may be smaller since the loss has a reduced

slope for such points. This leads to an *adaptive* regularization mechanism that may automatically reduce the amount of regularization when the data is easily separable. However, the robust optimization approach might only encourage local stability around training examples, while the global quantity $\|f\|_{\mathcal{H}}$ may become large in order to better fit the data. We note that a perfect fit of the data with large complexity does not prevent generalization (see, *e.g.*, Belkin et al., 2018a,b); yet, such mechanisms are still poorly understood. Nevertheless, it is easy to show that the robust objective (3.7) lower bounds the penalized objective with penalty $\epsilon\|f\|_{\mathcal{H}}$.

Gradient penalties. Taking $\bar{U} = \{\frac{\Phi(x)-\Phi(y)}{\|x-y\|_2} : x, y \in \mathcal{X}\}$, which is a subset of U by Eq. (3.2)—it turns out that this is the same set as for adversarial perturbation penalties, since Φ is homogeneous (Bietti and Mairal, 2019a) and $\mathcal{X} = \mathbb{R}^d$—we obtain a lower bound based on the Lipschitz constant of f:

$$\|f\|_{\mathcal{H}} \geq \sup_{x,y\in\mathcal{X}} \frac{f(x) - f(y)}{\|x - y\|_2} \geq \|\nabla f\| := \sup_{x\in\mathcal{X}} \|\nabla f(x)\|_2, \tag{3.8}$$

where the second inequality becomes an equality when $\mathcal{X}$ is convex, and the supremum is taken over points where f is differentiable. Although we are unaware of previous work using this exact lower bound for a generic regularization penalty, we note that variants replacing the supremum over x by an expectation over data have been recently used to stabilize the training of generative adversarial networks (Gulrajani et al., 2017; Roth et al., 2017), and we provide insights in Section 3.3.2 on the benefits of RKHS regularization in such a setting. Related penalties have been considered in the context of robust optimization, for regularization or robustness, noting that a penalty based on the gradient of the loss function $x \mapsto \ell(y, f(x))$ can give a good approximation of (3.7) when ϵ is small (Drucker and Le Cun, 1991; Lyu et al., 2015; Roth et al., 2018; Simon-Gabriel et al., 2019).

Penalties based on deformation stability. We may also obtain new penalties by considering more exotic sets $\bar{U} = \{\Phi(\tilde{x}) - \Phi(x) : x \in \mathcal{X},\ \tilde{x} \text{ is a small deformation of } x\}$, where the amount of deformation is dictated by the stability bounds of Chapter 2 in order to ensure that $\bar{U} \subset U$. More precisely, such bounds depend on the maximum displacement and Jacobian norm of the diffeomorphisms considered. These can be easily computed for various parameterized families of transformations, such as translations, scaling or rotations, leading to simple ways to control the regularization strength through the parameters of these transformations. One can also consider infinitesimal deformations from such parameterized transformations, which approximately yields the *tangent propagation* regularization strategy of Simard et al. (1998). These approaches are further detailed in Section 3.2.5 below. If instead we consider the robust optimization formulation (3.7), we obtain a form of *data augmentation* where transformations are optimized instead of sampled, as done by (Engstrom et al., 2019).

Extensions to multiple classes. We now extend the regularization strategies based on lower bounds to multi-valued networks, in order to deal with multiple classes. For that purpose, we consider a multi-class penalty $\|f_1\|_{\mathcal{H}}^2 + \ldots + \|f_K\|_{\mathcal{H}}^2$ for an $\mathbb{R}^K$-valued

function $f = (f_1, f_2, \ldots, f_K)$, and we define

$$\|f\|_\delta^2 := \sum_{k=1}^K \|f_k\|_\delta^2 \quad \text{and} \quad \|\nabla f\|^2 := \sum_{k=1}^K \|\nabla f_k\|^2,$$

where $\|f_k\|_\delta$ is the adversarial penalty (3.5), and $\|\nabla f_k\|$ is defined in (3.8). For deformation stability penalties, we proceed in a similar manner, and for robust optimization formulations (3.7), the extension is straightforward, given that multi-class losses such as cross-entropy can be directly optimized in an adversarial training or gradient penalty setup.

3.2.3 Exploiting upper bounds with spectral norms

Instead of lower bounds, one may use instead the following upper bound from Bietti and Mairal (2019a, Proposition 14):

$$\|f\|_{\mathcal{H}} \leq \omega(\|W_1\|, \ldots, \|W_L\|), \tag{3.9}$$

where ω is increasing in all of its arguments, and $\|W_k\|$ is the spectral norm of the linear operator W_k. Here, we simply consider the spectral norm on the filters, given by $\|W\| := \sup_{\|x\|_2 \leq 1} \|Wx\|_2$. Other generalization bounds relying on similar quantities have been proposed for controlling complexity (Bartlett et al., 2017; Neyshabur et al., 2018), suggesting that using them for regularization is relevant even beyond our kernel perspective, as observed by Cisse et al. (2017); Sedghi et al. (2019); Yoshida and Miyato (2017). Extensions to multiple classes are simple to obtain by simply considering spectral norms up to the last layer.

Penalizing the spectral norms. One way to control the upper bound (3.9) when learning a neural network f_θ is to consider a regularization penalty based on spectral norms

$$\min_\theta \frac{1}{n} \sum_{i=1}^n \ell(y_i, f_\theta(x_i)) + \lambda \sum_{l=1}^L \|W_l\|^2, \tag{3.10}$$

where λ is a regularization parameter. To optimize this cost, one can obtain (sub)gradients of the penalty by computing singular vectors associated to the largest singular value of each W_l. We consider the method of Yoshida and Miyato (2017), which computes such singular vectors approximately using one or two iterations of the power method, as well as a more costly approach using the full SVD.

Constraining the spectral norms with a continuation approach. In the constrained setting, we want to optimize:

$$\min_\theta \frac{1}{n} \sum_{i=1}^n \ell(y_i, f_\theta(x_i)) \;\; \text{s.t.} \;\; \|W_l\| \leq \tau \; ; \, l \in 1, \ldots, L \; ,$$

where τ is a user-defined constraint. This objective may be optimized by projecting each W_l in the spectral norm ball of radius τ after each gradient step. Such a projection is achieved by truncating the singular values to be smaller than τ. We found that the

loss was hardly optimized with this approach, and therefore introduce a continuation approach with an exponentially decaying schedule for τ reaching a constant τ_0 after a few epochs, which we found to be important for good empirical performance. More precisely, at epoch e we take

$$\tau = \tau_0(1 + \exp(-e/\kappa)),$$

and take κ to be 2 epochs for regularization, and 50 epochs for robustness. In the context of convolutional networks, we simply consider the singular values of a reshaped filter matrix, but we note that alternative approaches based on the singular values of the full convolutional operation may also be used (Sedghi et al., 2019).

3.2.4 Combining upper and lower bounds

One advantage of lower bound penalties is that they are independent of the model parameterization, making them flexible enough to use with more complex architectures. In addition, the connection with robust optimization can provide a useful mechanism for adaptive regularization. However, they do not provide a guaranteed control on the RKHS norm, unlike the upper bound strategies. This is particularly true for robust optimization approaches, which may favor small training loss and local stability over global stability through $\|f\|_{\mathcal{H}}$. Nevertheless, we observed that our new approaches based on separate penalties sometimes do help in controlling upper bounds as well (see Section 3.4).

While these upper bound strategies are useful for limiting model complexity, we found them empirically less effective for robustness (see Section 3.4.2). However, we observed that combining with lower bound approaches can overcome this weakness, perhaps due to a better control of local stability. In particular, such combined approaches often provide the best generalization performance in small data scenarios, as well as better guarantees on adversarially robust generalization thanks to a tighter control of the RKHS norm.

3.2.5 Deformation stability penalties

This section provides more details on the deformation stability penalties mentioned in Section 3.2.2, and the practical versions we use in our experiments of Section 3.4 on the Infinite MNIST dataset (Loosli et al., 2007).

Stability to deformations. We begin by providing some background on deformation stability, recalling that these can provide new lower bound penalties as explained in Section 3.2.2. Viewing an element $x \in \mathcal{X}$ as a signal $x(u)$, where u denotes the location (*e.g.* a two-dimensional vector for images), we denote by x_τ a deformed version of x given by $x_\tau(u) = x(u - \tau(u))$, where τ is a diffeomorphism. The deformation stability bounds derived in Chapter 2 take the form:

$$\|\Phi(x_\tau) - \Phi(x)\|_{\mathcal{H}} \leq (C_1\|\tau\|_\infty + C_2\|\nabla\tau\|_\infty)\|x\|, \tag{3.11}$$

where $\nabla\tau(u)$ is the Jacobian of τ at location u. Here, C_1 controls translation invariance and typically decreases with the total amount of pooling (*i.e.*, translation invariance more or less corresponds to the resolution at the final layer), while C_2 controls stability to deformations (note that $\nabla\tau = 0$ for translations) and is typically smaller when using

small patches. We note that the bounds assume linear pooling layers with a certain spatial decay, adapted to the resolution of the current layer; our experiments in Section 3.4 on Infinite MNIST with deformation stability penalties thus use average pooling layers on 2x2 neighborhoods.

Adversarial deformation penalty. We can obtain lower bound penalties by exploiting the above stability bounds in a similar manner to the adversarial perturbation penalty introduced in Section 3.2.2. In particular, assuming a scalar-valued convolutional network f:

$$\|f\|_\tau^2 := \sup_{x\in\mathcal{X},\tau\in\mathcal{T}} (f(x_\tau) - f(x))^2 \tag{3.12}$$

where $\mathcal{T}$ is a collection of diffeomorphisms. When the diffeomorphisms in $\mathcal{T}$ have bounded norm $\|\tau\|_\infty$ and Jacobian norm $\|\nabla\tau\|_\infty$, and assuming $\mathcal{X}$ (or, in practice, the training data) is bounded, the stability bound (3.11) ensures that the set $U_\mathcal{T} = \{\Phi(x_\tau) - \Phi(x) : x \in \mathcal{X}, \tau \in \mathcal{T}\}$ is included in an RKHS ball with some radius r, so that $\|f\|_\tau$ is a lower bound on $r\|f\|_\mathcal{H}$.

Tangent gradient penalty. We also consider the following gradient penalty along tangent vectors, which provides an approximation of the above adversarial penalty when considering small, parameterized deformations, and recovers the tangent propagation strategy of Simard et al. (1998):

$$\|D_\tau f\|^2 := \sup_{x\in\mathcal{X}} \|\partial_\alpha f(x + \sum_i \alpha_i t_{x,i})\|^2, \tag{3.13}$$

where $\{t_{x,i}\}_{i=1,\dots,q}$ are tangent vectors at x obtained from a given set of deformations. To see the link with the adversarial deformation penalty (3.12), consider for simplicity a single deformation, $\mathcal{T} = \{\tau_0\}$. For small α, we have

$$x_{\alpha\tau_0} \approx x + \alpha t_x, \quad \text{where} \quad t_x(u) = \tau_0(u) \cdot \nabla x(u),$$

where t_x denotes the tangent vector of the deformation manifold $\{\alpha\tau_0 : \alpha\}$ at $\alpha = 0$ (Simard et al., 1998). Then,

$$f(x_{\alpha\tau_0}) - f(x) \approx \alpha\partial_\alpha f(x + \alpha t_x) = \alpha\langle\nabla f(x), t_x\rangle.$$

In this case, denoting $\alpha\mathcal{T} = \{\alpha\tau_0\}$, we have

$$\sup_{x\in\mathcal{X},\tau\in\alpha\mathcal{T}} (f(x_\tau) - f(x))^2 \approx \alpha^2 \sup_{x\in\mathcal{X}} |\partial_\alpha f(x + \alpha t_x)|^2,$$

so that when α is small, the adversarial penalty can be approximated by $\alpha\|D_\tau f\|$ (note that using $\alpha\mathcal{T}$ instead of $\mathcal{T}$ in the adversarial penalty would also yield a scaling by α, since the stability bounds imply α times smaller perturbations in the RKHS).

Practical implementations on Infinite MNIST. In our experiments on Infinite MNIST in Section 3.4, we compute $\|f\|_\tau^2$ by considering 32 random transformations of each digit in a mini-batch of training examples, and taking the maximum over both the example and the transformation. We do this separately for each class, as for the other lower bound penalties $\|f\|_\delta^2$ and $\|\nabla f\|^2$. For $\|D_\tau f\|^2$, we take $\{t_{x,i}\}_{i=1,\dots,q}$ with $q = 30$ to be tangent vectors given by random diffeomorphisms from Infinite MNIST around each example x.

3.2.6 Extensions to Non-Euclidian Geometries

The kernel approach from previous sections is well-suited for input spaces $\mathcal{X}$ equipped with the Euclidian distance, thanks to the non-expansiveness property (3.2) of the kernel mapping. In the case of linear models, this kernel approach corresponds to using ℓ_2-regularization by taking a linear kernel. However, other forms of regularization and geometries can often be useful, for example to encourage sparsity with an ℓ_1 regularizer. Such a regularization approach presents tight links with robustness to ℓ_∞ perturbations on input data, thanks to the duality relation $\|w\|_1 = \sup_{\|u\|_\infty} \langle w, u \rangle$ (see Xu et al., 2009a).

In the context of deep networks, we can leverage such insights to obtain new regularizers, expressed in the same variational form as the lower bounds in Section 3.2.2, but with different geometries on $\mathcal{X}$. For ℓ_∞ perturbations, we obtain

$$\sup_{x,y \in \mathcal{X}} \frac{f(x) - f(y)}{\|x - y\|_\infty} \quad \geq \quad \sup_{x \in \mathcal{X}} \|\nabla f(x)\|_1. \tag{3.14}$$

The Lipschitz regularizer (l.h.s.) can also be taken in an adversarial perturbation form, with ℓ_∞-bounded perturbations $\|\delta\|_\infty \leq \epsilon$. When considering the corresponding robust optimization problem

$$\min_\theta \frac{1}{n} \sum_{i=1}^{n} \sup_{\|\delta\|_\infty \leq \epsilon} \ell(y_i, f_\theta(x_i + \delta)), \tag{3.15}$$

we may consider the PGD approach of Madry et al. (2018), or the associated gradient penalty approach with the ℓ_1 norm, which is a good approximation when ϵ is small (Lyu et al., 2015; Simon-Gabriel et al., 2019).

As most visible in the gradient ℓ_1-norm in (3.14), these penalties encourage some sparsity in the gradients of f, which is a reasonable prior for regularization on images, for instance, where we might only want predictions to change based on few salient pixel regions. This can lead to gains in interpretability, as observed by Tsipras et al. (2019).

3.3 Theoretical Guarantees and Insights

In this section, we study how the kernel perspective allows us to extend standard margin-based generalization bounds to an adversarial setting in order to provide theoretical guarantees on adversarially robust generalization. We then discuss how our kernel approach provides novel interpretations for training generative adversarial networks.

3.3.1 Guarantees on adversarial generalization

While various methods have been introduced to empirically gain robustness to adversarial perturbations, the ability to generalize with such perturbations, also known as *adversarial generalization* (Schmidt et al., 2018), still lacks theoretical understanding. Margin-based bounds have been useful to explain the generalization behavior of learning algorithms that can fit the training data well, such as kernel methods, boosting and neural networks (Koltchinskii and Panchenko, 2002; Boucheron et al., 2005; Bartlett et al., 2017). Here, we show how such arguments can be adapted to obtain guarantees on adversarial generalization, *i.e.*, on the expected classification error in the presence of an ℓ_2-bounded adversary, based on the RKHS norm of a learned model. For a binary

classification task with labels in $\mathcal{Y} = \{-1, 1\}$ and data distribution $\mathcal{D}$, we would like to bound the expected adversarial error of a classifier f, given for some $\epsilon > 0$ by

$$\mathrm{err}_{\mathcal{D}}(f, \epsilon) := P_{(x,y)\sim\mathcal{D}}(\exists \|\delta\|_2 \leq \epsilon : \; yf(x+\delta) < 0). \tag{3.16}$$

Leveraging the fact that f is $\|f\|_{\mathcal{H}}$-Lipschitz, we now show how to further bound this quantity using empirical margins, following the usual approach to obtaining margin bounds for kernel methods (*e.g.*, Boucheron et al., 2005). Consider a training dataset $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$. Defining $L_n^{\gamma}(f) := \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{y_i f(x_i) < \gamma\}$, we have the following bound:

Proposition 3.1 (Adversarially robust margin bound)**.** *With probability $1-\delta$ over a dataset $\{(x_i, y_i)\}_{i=1,\ldots,n}$, we have, for all choices of $\gamma > 0$ and $f \in \mathcal{H}$,*

$$err_{\mathcal{D}}(f, \epsilon) \leq L_n^{\gamma + 2\epsilon\|f\|_{\mathcal{H}}}(f) + \tilde{O}\left(\frac{\|f\|_{\mathcal{H}}\bar{B}}{\gamma\sqrt{n}}\right), \tag{3.17}$$

where $\bar{B} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} K(x_i, x_i)}$ and $\tilde{O}$ hides a term depending logarithmically on $\|f\|_{\mathcal{H}}, \gamma$, and δ.

Proof. Assume for now that γ is fixed in advance, and let $\mathcal{F}_\lambda := \{f \in \mathcal{H} : \|f\|_{\mathcal{H}} \leq \lambda\}$. Note that for all $f \in \mathcal{F}_\lambda$ we have

$$\mathrm{err}_{\mathcal{D}}(f, \epsilon) = P(\exists \|\delta\| \leq \epsilon : yf(x+\delta) < 0) \leq P(yf(x) < \lambda\epsilon) =: L^{\lambda\epsilon}(f),$$

since $\|f\|_{\mathcal{H}} \leq \lambda$ is an upper bound on the Lipschitz constant of f. Consider the function

$$\phi(x) = \begin{cases} 0, & \text{if } x \leq -\gamma - \lambda\epsilon \\ 1, & \text{if } x \geq -\lambda\epsilon \\ 1 + (x + \lambda\epsilon)/\gamma, & \text{otherwise.} \end{cases}$$

Defining $A(f) = \mathbb{E}\,\phi(-yf(x)) \geq L^{\lambda\epsilon}(f)$ and $A_n(f) = \frac{1}{n}\sum_{i=1}^{n}\phi(-y_i f(x_i)) \leq L_n^{\lambda\epsilon+\gamma}(f)$, and noting that ϕ is upper bounded by 1 and $1/\gamma$ Lipschitz, we can apply similar arguments to (Boucheron et al., 2005, Theorem 4.1) to obtain, with probability $1-\delta$,

$$L^{\lambda}\epsilon(f) \leq L_n^{\lambda\epsilon+\gamma}(f) + O\left(\frac{1}{\gamma}R_n(\mathcal{F}_\lambda) + \sqrt{\frac{\log 1/\delta}{n}}\right),$$

where $R_n(\mathcal{F}_\lambda)$ denotes the empirical Rademacher complexity of $\mathcal{F}_\lambda$ on the dataset $\{(x_i, y_i)\}_{i=1,\ldots,n}$. Standard upper bounds on empirical Rademacher complexity of kernel classes with bounded RKHS norm yield the following bound

$$\mathrm{err}_{\mathcal{D}}(f, \epsilon) \leq L_n^{\lambda\epsilon+\gamma}(f) + O\left(\frac{\lambda}{\gamma\sqrt{n}}\sqrt{\frac{1}{n}\sum_{i=1}^{n} K(x_i, x_i)} + \sqrt{\frac{\log 1/\delta}{n}}\right).$$

Note that the bound is still valid with $\gamma' \geq \gamma$ instead of γ in the first term of the r.h.s., since $L_n^{\gamma}(f)$ is non-decreasing as a function of γ.

In order to establish the final bound, we instantiate the previous bound for values $\lambda_i = 2^i$ and $\gamma_j = 2^{-j}$. Defining $\delta_{i,j} = \frac{\delta}{(1+4i^2)\cdot(1+4j^2)}$, we have that w.p. $1 - \delta_{i,j}$, for all $f \in \mathcal{F}_{\lambda_i}$ and all $\gamma \geq \gamma_j$,

$$\mathrm{err}_{\mathcal{D}}(f, \epsilon) \leq L_n^{\lambda_i \epsilon + \gamma}(f) + O\left(\frac{\lambda_i}{\gamma_j \sqrt{n}} \sqrt{\frac{1}{n}\sum_{i=1}^{n} K(x_i, x_i)} + \sqrt{\frac{\log 1/\delta_{i,j}}{n}}\right). \tag{3.18}$$

By a union bound, this event holds jointly for all integers i, j w.p. greater than $1 - \delta$, since $\sum_{i,j} \delta_{i,j} \leq \delta$. Now consider an arbitrary $f \in \mathcal{H}$ and $\gamma > 0$ and let $i = \lceil \log_2 \|f\|_{\mathcal{H}} \rceil$ and $j = \lceil \log_2(1/\gamma) \rceil$. We have

$$\begin{aligned} \lambda_i &\leq 2\|f\|_{\mathcal{H}} \\ \frac{1}{\gamma_j} &\leq \frac{2}{\gamma} \\ \log(1/\delta_{i,j}) &\leq \log(C(\|f\|_{\mathcal{H}}, \gamma)/\delta), \end{aligned}$$

with $C(\|f\|_{\mathcal{H}}, \gamma) := (1 + 4(\log_2 \|f\|_{\mathcal{H}})^2) \cdot (1 + 4(\log_2(1/\gamma))^2)$. Applying this to the bound in (3.18) yields the desired result.

□

When $\epsilon = 0$, we obtain the usual margin bound, while $\epsilon > 0$ yields a bound on adversarial error $\mathrm{err}_{\mathcal{D}}(f, \epsilon)$, for some neural network f learned from data. Note that other complexity measures based on products of spectral norms may be used instead of $\|f\|_{\mathcal{H}}$, as well as multi-class extensions, following Bartlett et al. (2017); Neyshabur et al. (2018). In concurrent work, Khim and Loh (2018); Yin et al. (2019) derive similar bounds in the context of fully-connected networks. In contrast to these works, which bound complexity of a modified function class, our bound uses the complexity of the original class and leverages smoothness properties of functions to derive the margin bound.

One can then study the effectiveness of a regularization algorithm by inspecting cumulative distribution (CDF) plots of the *normalized margins* $\bar{\gamma}_i = y_i f(x_i)/\|f\|_{\mathcal{H}}$, for different strengths of regularization (an example is given in Figure 3.2, Section 3.4.2). According to the bound (3.17), one can assess expected adversarial error with ϵ-bounded perturbations by looking at the part of the plot to the right of $\bar{\gamma} = 2\epsilon$. In particular, the value of the CDF at such a value of $\bar{\gamma}$ is representative of the bound for large n (since the second term is negligible), while for smaller n, the best bound is obtained for a larger value of $\bar{\gamma}$, which also suggests that the right side of the plots is indicative of performance on small datasets.

When the RKHS norm can be well approximated, our bound provides a certificate on test error in the presence of adversaries. While such an approximation is difficult to obtain in general, the guarantee is most useful when lower and upper bounds of the RKHS norm are controlled together.

We note that in the case of linear models, our robust margin bound can be adapted to ℓ_∞-perturbations, by leveraging Rademacher complexity bounds for ℓ_1-constrained models (Kakade et al., 2009). Extensions to deeper networks are possible (see Khim and Loh, 2018; Yin et al., 2019).

3.3.2 New insights on generative adversarial networks

Generative adversarial networks (GANs) attempt to learn a *generator* neural network $G_\phi : \mathcal{Z} \to \mathcal{X}$, so that the distribution of $G_\phi(z)$ with $z \sim D_z$ a noise vector resembles a data distribution D_x. In this section, we discuss connections between recent regularization techniques for training GANs, and approaches to learning generative models based on a MMD criterion (Gretton et al., 2012), in view of our RKHS framework. Our goal is to provide a new insight on these methods, but not necessarily to provide a new one.

Various recent approaches have relied on regularization strategies on a *discriminator* network in order to improve the stability of GAN training and the quality of the produced samples. Some of these resemble the approaches presented in Section 3.2 such as gradient penalties (Gulrajani et al., 2017; Roth et al., 2017) and spectral norm regularization (Miyato et al., 2018a). We provide an RKHS interpretation of these methods as optimizing an MMD distance with the convolutional kernel introduced in Section 3.2:

$$\min_{\phi} \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{x \sim D_x}[f(x)] - \mathbb{E}_{z \sim D_z}[f(G_\phi(z))]. \tag{3.19}$$

When learning from an empirical distribution over n samples, the MMD criterion is known to have much better sample complexity than the Wasserstein-1 distance considered by Arjovsky et al. (2017) for high-dimensional data such as images (Sriperumbudur et al., 2012). While the MMD approach has been used for training generative models, it generally relies on a generic kernel function, such as a Gaussian kernel, that appears explicitly in the objective (Dziugaite et al., 2015; Li et al., 2017; Bińkowski et al., 2018). Although using a learned feature extractor can improve this, the Gaussian kernel might be a poor choice when dealing with natural signals such as images, while the hierarchical kernel we consider in this work is better suited for this type of data, by providing useful invariance and stability properties. Leveraging the variational form of the MMD (3.19) with this kernel suggests for instance using convolutional networks as the discriminator f, with constraints on the spectral norms in order to ensure $\|f\|_{\mathcal{H}} \leq C$ for some C, as done by Miyato et al. (2018a) through normalization.

3.4 Experiments

We tested the regularization strategies presented in Section 3.2 in the context of improving generalization on small datasets and training robust models. Our goal is to use common architectures used for large datasets and improve their performance in different settings through regularization. Our Pytorch implementation of the various strategies is available at `https://github.com/albietz/kernel_reg`.

For the adversarial training strategies, the inner maximization problems are solved using 5 steps of projected gradient ascent with constant step-lengths. In the case of the lower bound penalties $\|f\|_\delta^2$ and $\|\nabla f\|^2$, we also maximize over examples in the mini-batch, only considering the maximal element when computing gradients with respect to parameters. For the robust optimization problem (3.7), we use PGD with ℓ_2 perturbations, as well as the corresponding ℓ_2 (squared) gradient norm penalty on the loss. For the upper bound approaches with spectral norms (SNs), we consider the SN projection strategy with decaying τ, as well as the SN penalty (3.10), either using power iteration (PI) or a full SVD for computing gradients.

Table 3.1: Regularization on CIFAR10 with 1 000 examples for VGG-11 and ResNet-18. Each entry shows the test accuracy with/without data augmentation when all hyper-parameters are optimized on a validation set. See also Section 3.A.1 in the appendix for additional results and statistical testing.

Method	1k VGG-11	1k ResNet-18
No weight decay	50.70 / 43.75	45.23 / 37.12
Weight decay	51.32 / 43.95	44.85 / 37.09
SN penalty (PI)	54.64 / 45.06	47.01 / 39.63
SN projection	54.14 / **46.70**	47.12 / 37.28
VAT	50.88 / 43.36	47.47 / 42.82
PGD-ℓ_2	51.25 / 44.40	45.80 / 41.87
grad-ℓ_2	**55.19** / 43.88	**49.30** / **44.65**
$\|f\|_\delta^2$ penalty	51.41 / 45.07	48.73 / 43.72
$\|\nabla f\|^2$ penalty	54.80 / 46.37	**48.99** / **44.97**
PGD-ℓ_2 + SN proj	54.19 / **46.66**	47.47 / 41.25
grad-ℓ_2 + SN proj	**55.32** / **46.88**	48.73 / 42.78
$\|f\|_\delta^2$ + SN proj	54.02 / **46.72**	48.12 / 43.56
$\|\nabla f\|^2$ + SN proj	**55.24** / **46.80**	**49.06** / **44.92**

3.4.1 Improving generalization on small datasets

We consider the datasets CIFAR10 and MNIST when using a small number of training examples, as well as 102 datasets of biological sequences that suffer from small sample size.

CIFAR10. In this setting, we use 1 000 and 5 000 examples of the CIFAR10 dataset, with or without data augmentation. We consider a VGG network (Simonyan and Zisserman, 2014) with 11 layers, as well as a residual network (He et al., 2016) with 18 layers, which achieve 91% and 93% test accuracy respectively when trained on the full training set with standard data augmentation (horizontal flips + random crops). We do not use batch normalization layers in order to prevent any interaction with spectral norms. Each strategy derived in Section 3.2 is trained for 500 epochs using SGD with momentum and batch size 128, halving the step-size every 40 epochs. In order to study the potential effectiveness of each method, we assume that a reasonably large validation set is available to select hyper-parameters; thus, we keep 10 000 annotated examples for this purpose. We also show results using a smaller validation set in Appendix 3.A.1.

Table 3.1 shows the test accuracies on 1 000 examples for upper and lower bound approaches, as well as combined ones. We also include virtual adversarial training (VAT, Miyato et al., 2018b). We provide extended tables in Appendix 3.A.1 with additional methods, other geometries, results for 5 000 examples, as well as hypothesis tests for comparing pairs of methods and assessing the significance of our findings. Overall, we find that the combined lower bound + SN constraints approaches often yield better results than either method separately. For lower bound approaches alone, we found our $\|f\|_\delta^2$ and $\|\nabla f\|^2$ penalties to often work best, particularly without data augmentation, while robust optimization strategies can be preferable with data augmentation, perhaps thanks to the adaptive regularization effect discussed earlier, which may be helpful in this easier setting. Gradient penalties often outperform adversarial perturbation strate-

Table 3.2: Regularization on 300 or 1 000 examples from MNIST, using deformations from Infinite MNIST. $(*)$ indicates that random deformations were included as training examples, while $\|f\|_\tau^2$ and $\|D_\tau f\|^2$ use them as part of the regularization penalty. See Section 3.A.2 in the appendix for more results and statistical testing.

Method	300 VGG	1k VGG
Weight decay	89.32	94.08
SN projection	90.69	95.01
grad-ℓ_2	93.63	96.67
$\|f\|_\delta^2$ penalty	94.17	96.99
$\|\nabla f\|^2$ penalty	94.08	96.82
Weight decay $(*)$	92.41	95.64
grad-ℓ_2 $(*)$	95.05	97.48
$\|D_\tau f\|^2$ penalty	94.18	96.98
$\|f\|_\tau^2$ penalty	94.42	97.13
$\|f\|_\tau^2 + \|\nabla f\|^2$	94.75	97.40
$\|f\|_\tau^2 + \|f\|_\delta^2$	95.23	**97.66**
$\|f\|_\tau^2 + \|f\|_\delta^2$ $(*)$	**95.53**	**97.56**
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj	95.20	**97.60**
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj $(*)$	**95.40**	**97.77**

gies, possibly because of the closed form gradients which may improve optimization. We also found that adversarial training strategies tend to poorly control SNs compared to gradient penalties, particularly PGD (see also Section 3.4.2). SN constraints alone can also work well in some cases, particularly for VGG architectures, and often outperform SN penalties. SN penalties can work well nevertheless and provide computational benefits when using the power iteration variant.

Infinite MNIST. In order to assess the effectiveness of lower bound penalties based on deformation stability, we consider the Infinite MNIST dataset (Loosli et al., 2007), which provides an "infinite" number of transformed generated examples for each of the 60 000 MNIST training digits. Here, we use a 5-layer VGG-like network with average pooling after each 3x3 convolution layer, in order to more closely match the architecture assumptions of Chapter 2 for deformation stability. We consider two lower bound penalties that leverage the digit transformations in Infinite MNIST: one based on "adversarial" deformations around each digit, denoted $\|f\|_\tau^2$; and a tangent propagation (Simard et al., 1998) variant, denoted $\|D_\tau f\|^2$, which provides an approximation to $\|f\|_\tau^2$ for small deformations based on gradients along a few tangent vector directions given by deformations (see Section 3.2.5 for details). Table 3.2 shows the obtained test accuracy for subsets of MNIST of size 300 and 1 000. Overall, we find that combining both adversarial penalties $\|f\|_\tau^2$ and $\|f\|_\delta^2$ performs best, which suggests that it is helpful to obtain tighter lower approximations of the RKHS norm by considering perturbations of different kinds. Explicitly controlling the spectral norms can further improve performance, as does training on deformed digits, which may yield better margins by exploiting the additional knowledge that small deformations preserve labels. Note that data augmentation alone (with some weight decay) does quite poorly in this case, even compared to our lower bound penalties which do not use deformations.

Table 3.3: Regularization on protein homology detection tasks, with or without data augmentation (DA). Fixed hyperparameters are selected using the first half of the datasets, and we report the average auROC50 score on the second half. See Section 3.A.3 in the appendix for more details and statistical testing.

Method	No DA	DA
No weight decay	0.421	0.541
Weight decay	0.432	0.544
SN proj	0.583	0.615
PGD-ℓ_2	0.488	0.554
grad-ℓ_2	0.551	0.570
$\|f\|_\delta^2$	0.577	0.611
$\|\nabla f\|^2$	0.566	0.598
PGD-ℓ_2 + SN proj	**0.615**	**0.622**
grad-ℓ_2 + SN proj	0.581	**0.634**
$\|f\|_\delta^2$ + SN proj	**0.631**	**0.639**
$\|\nabla f\|^2$ + SN proj	0.576	**0.617**

Protein homology detection. Remote homology detection between protein sequences is an important problem to understand protein structure. Given a protein sequence, the goal is to predict whether it belongs to a superfamily of interest. We consider the Structural Classification Of Proteins (SCOP) version 1.67 dataset (Murzin et al., 1995), which we process as described in Appendix 3.A.3 in order to obtain 102 balanced binary classification tasks with 100 protein sequences each, thus resulting in a low-sample regime. Protein sequences were also cut to 400 amino acids.

Sequences are represented with a one-hot encoding strategy—that is, a sequence of length l is represented as a binary matrix in $\{0,1\}^{20\times l}$, where 20 is the number of different amino acids (alphabet size of the sequences). Such a structure can then be processed by convolutional neural networks (Alipanahi et al., 2015). In this section, we do not try to optimize the structure of the network for the task, since our goal is only to evaluate the effect of regularization strategies. Therefore, we use a simple convolutional network with 3 convolutional layers followed by global max-pooling and a final fully-connected layer (we use filters of size 5, and a max-pooling layer after the second convolutional layer).

Training was done using Adam with a learning rate fixed to 0.01, and a weight decay parameter tuned for each method. Since hyper-parameter selection per dataset is difficult due to the low sample size, we use the same parameters across datasets. This allows us to use the first 51 datasets as a validation set for hyper-parameter tuning, and we report average performance with these fixed choices on the remaining 51 datasets. The standard performance measure for this task is the auROC50 score (area under the ROC curve up to 50% false positives). We note that the selection of hyper-parameters has a transductive component, since some of the sequences in the test datasets may also appear in the datasets used for validation (possibly with a different label).

The results are shown in Table 3.3. The procedure used for data augmentation (right column) is described in Appendix 3.A.3. We found that the most effective approach is the adversarial perturbation penalty, together with SN constraints. In particular, we found it to outperform the gradient penalty $\|\nabla f\|^2$, perhaps because in this case

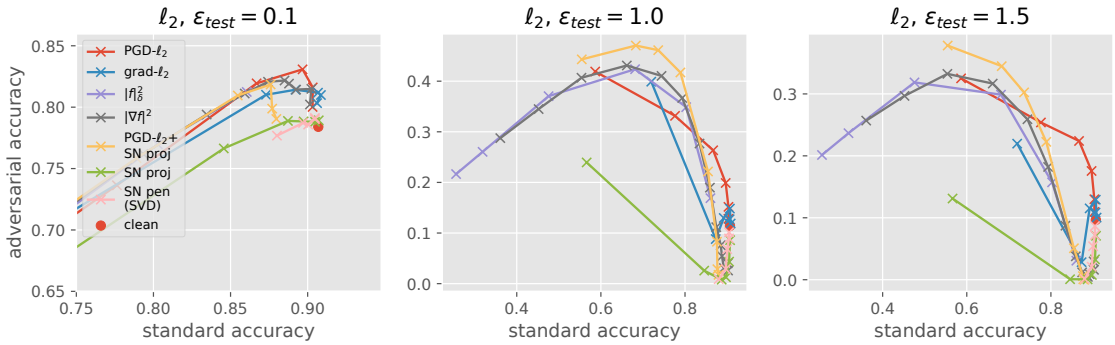


Figure 3.1: Robustness trade-off curves of different regularization methods for VGG11 on CIFAR10. Each plot shows test accuracy vs adversarial test accuracy for ℓ_2-bounded, 100-step PGD adversaries with a fixed ϵ_{test}. Different points on a curve correspond to training with different regularization strengths. The regularization increases monotonically along a given curve, and the leftmost points correspond to the strongest regularization. For PGD-ℓ_2 + SN projection, we vary ϵ with a fixed $\tau = 0.8$.

gradient penalties are only computed on a discrete set of possible points given by one-hot encodings, while adversarial perturbations may increase stability to wider regions, potentially covering different possible encoded sequences.

3.4.2 Training adversarially robust models

We consider the same VGG architecture as in Section 3.4.1, trained on CIFAR10 with data augmentation, with different regularization strategies. Each method is trained for 300 epochs using SGD with momentum and batch size 128, dividing the step-size in half every 30 epochs. This strategy was successful in reaching convergence for all methods.

Figure 3.1 shows the test accuracy of the different methods in the presence of ℓ_2-bounded adversaries, plotted against standard accuracy. We can see that the robust optimization approaches tend to work better in high-accuracy regimes, perhaps because the local stability that they encourage is sufficient on this dataset, while the $\|f\|_\delta^2$ penalty can be useful in large-perturbation regimes. We find that upper bound approaches alone do not provide robust models, but combining the SN constraint approach with a lower bound strategy (in this case PGD-ℓ_2) helps improve robustness perhaps thanks to a more explicit control of stability. We note that our best performing method (in this case, the combined approach PGD-ℓ_2 with $\epsilon = 2.0$ + SN constraint with $\tau = 0.8$) achieves **47.04%** robust test accuracy for a 100-step PGD adversary with $\epsilon_{\text{test}} = 1.0$, which surpasses the best reported accuracies for ℓ_2-bounded adversaries with such ϵ_{test}, *e.g.*, in Rony et al. (2019); Salman et al. (2019), which are below 40% (though Salman et al. (2019) also provides certification through randomized smoothing, while our method is not certified). Hence, this provides state-of-the-art results for empirical robustness on this dataset, though we note that there are differences in the experimental setup, including the choice of model and attacks, which may also partially contribute to this gap.

The plots also confirm that gradient penalties on the loss may be preferable for small regularization strengths (they achieve higher accuracy while improving robustness for small ϵ_{test}), while for stronger regularization, the gradient approximation no longer holds

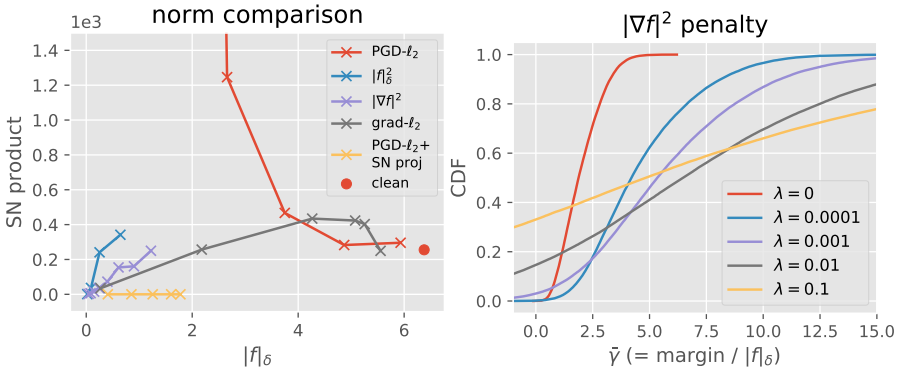


Figure 3.2: (left) Comparison of lower and upper bound quantities ($\|f\|_\delta$ vs the product of spectral norms). (right) CDF plot of normalized empirical margins for the $\|\nabla f\|^2$ penalty with different regularization strengths, normalized by $\|f\|_\delta$. We consider 1000 fixed training examples when computing $\|f\|_\delta$.

and the multi-step robust optimization approaches such as PGD (and its combination with SN constraints) are preferred. More experiments confirming these findings are available in Section 3.A.4 of the appendix.

Norm comparison and adversarial generalization. Figure 3.2 (left) compares lower and upper bound quantities for different regularization strengths. Note that for PGD, in contrast to other methods, we can see that the product of spectral norms (representative of an upper bound on $\|f\|_\mathcal{H}$) increases when the lower bound $\|f\|_\delta$ decreases. This suggests that a network learned with PGD with large ϵ may have large RKHS norm, possibly because the approach tries to separate ϵ-balls around the training examples, which may require a more complex model than simply separating the training examples (see also Madry et al., 2018). This large discrepancy between upper and lower bounds highlights the fact that such models may only be stable locally near training data, though this happens to be enough for robustness on many test examples on CIFAR10.

In contrast, for other methods, and in particular the lower bound penalties $\|f\|_\delta^2$ and $\|\nabla f\|^2$, the upper and lower bounds appear more tightly controlled, suggesting a more appropriate control of the RKHS norm. This makes our guarantees on adversarial generalization more meaningful, and thus we may look at the empirical distributions of normalized margins $\bar{\gamma}$ obtained using $\|f\|_\delta$ for normalization (as an approximation of $\|f\|_\mathcal{H}$), shown in Figure 3.2 (right). The curves suggest that for small $\bar{\gamma}$, and hence small ϵ_{test}, smaller values of λ are preferred, while stronger regularization helps for larger $\bar{\gamma}$, yielding lower test error guarantees in the presence of stronger adversaries according to our bounds in Section 3.3.1. This qualitative behavior is indeed observed in the results of Figure 3.1 on test data for the $\|\nabla f\|^2$ penalty.

Appendix

3.A Additional Experiment Results

3.A.1 CIFAR10

This section provides more extensive results for the experiments on CIFAR10 from Section 3.4.1. In particular, Table 3.A.1 shows additional experiments on larger subsets of size $5\dot{0}00$, as well as more methods, including different geometries (see Section 3.2.6). The table also reports results obtained when using a smaller validation set of size 1 000. The full hyper-parameter grid is given in Table 3.A.3.

In order to assess the statistical significance of our results, we repeated the experiments on 10 new random choices of subsets, using the hyperparameters selected on the original subset from Table 3.A.1 (except for learning rate, which is selected according to a different validation set for each subset). We then compared pairs of methods using a paired t-test, with p-values shown in Table 3.A.2. In particular, the results strengthen some of our findings, for instance, that $\|\nabla f\|^2$ should be preferred to the gradient penalty on the loss when there is no data augmentation, and that combined upper+lower bound approaches tend to outperform the individual upper or lower bound strategies.

3.A.2 Infinite MNIST

We provide more extensive results for the Infinite MNIST dataset in Table 3.A.4, in particular showing more regularization strategies, as well as results with or without data augmentation, marked with $(*)$. As in the case of CIFAR10, we use SGD with momentum (fixed to 0.9) for 500 epochs, with initial learning rates in $[0.005; 0.05; 0.5]$, and divide the step-size by 2 every 40 epochs. The full hyper-parameter grid is given in Table 3.A.6.

As in the case of CIFAR10, we report statistical significance tests in Table 3.A.5 comparing pairs of methods based on 10 different random choices of subsets. In particular, the results confirm that weight decay with data augmentation alone tends to give weaker results than separate penalties, and that the combined penalty $\|f\|_\tau^2 + \|f\|_\delta^2$, which combines adversarial perturbations of two different types, outperforms each penalty taken by itself on a single type of perturbation, which emphasizes the benefit of considering perturbations of different natures, perhaps thanks to a tighter lower bound approximation of the RKHS norm. We note that grad-$\ell_2(*)$ worked well on some subsets, but poorly on others due to training instabilities, possibly because of the selected hyperparameters which are quite large (and thus likely violate the approximation to the robust optimization objective).

3.A.3 Protein homology detection

Dataset description. Our protein homology detection experiments consider the Structural Classification Of Proteins (SCOP) version 1.67 dataset (Murzin et al., 1995), filtered and split following the procedures of Håndstad et al. (2007). Specifically, positive training samples are extracted from one superfamily from which one family is withheld to serve as positive test set, while negative sequences are chosen from outside of the target family's hold and are randomly split into training and test samples in the same ratio as positive samples. This yields 102 superfamily classification tasks, which are generally very class-imbalanced. For each task, we sample 100 class-balanced training samples to use as training set. The positive samples are extended to 50 with Uniref50 using PSI-BLAST (Altschul et al., 1997) if they are fewer.

Data augmentation procedure. We consider in our experiments a discrete way of perturbing training samples to perform data augmentation. Specifically, for a given sequence, a perturbed sequence can be obtained by randomly changing some of the characters. Each character in the sequence is switched to a different one, randomly chosen from the alphabet, with some probability p. We fixed this probability to 0.1 throughout the experiments.

Experimental details and significance tests. In our experiments, we use the Adam optimization algorithm with a learning rate fixed to 0.01 (and β fixed to defaults $(0.9, 0.999)$), with a batch size of 100 for 300 epochs. The full hyper-parameter grid is given in Table 3.A.8. In addition to the average auROC50 scores reported in Table 3.3, we perform paired t-tests for comparing pairs of methods in Table 3.A.7 in order to verify the significance of our findings. The results confirm that the adversarial perturbation penalty and its combination with spectral norm constraints tends to outperform the other approaches.

3.A.4 Robustness

Figure 3.A.1 extends Figure 3.1 from Section 3.4.2 to show more methods, adversary strenghts, and different geometries. For combined (PGD-ℓ_2 + SN projection) approaches, we can see that stronger constraints (*i.e.*, smaller τ) tend to reduce standard accuracy, likely because it prevents a good fit of the data, but can provide better robustness to strong adversaries ($\epsilon_{test} = 1$). We can see that using the right metric in PGD indeed helps against an ℓ_∞ adversary, nevertheless controlling global stability through the RKHS norm as in the $\|f\|_\delta^2$ and $\|\nabla f\|^2$ penalties can still provide some robustness against such adversaries, even with large ϵ_{test}. For gradient penalties, we find that the different geometries behave quite similarly, which may suggest that more appropriate optimization algorithms than SGD could be needed to better accommodate the non-smooth case of ℓ_1/ℓ_∞, or perhaps that both algorithms are actually controlling the same notion of complexity on this dataset.

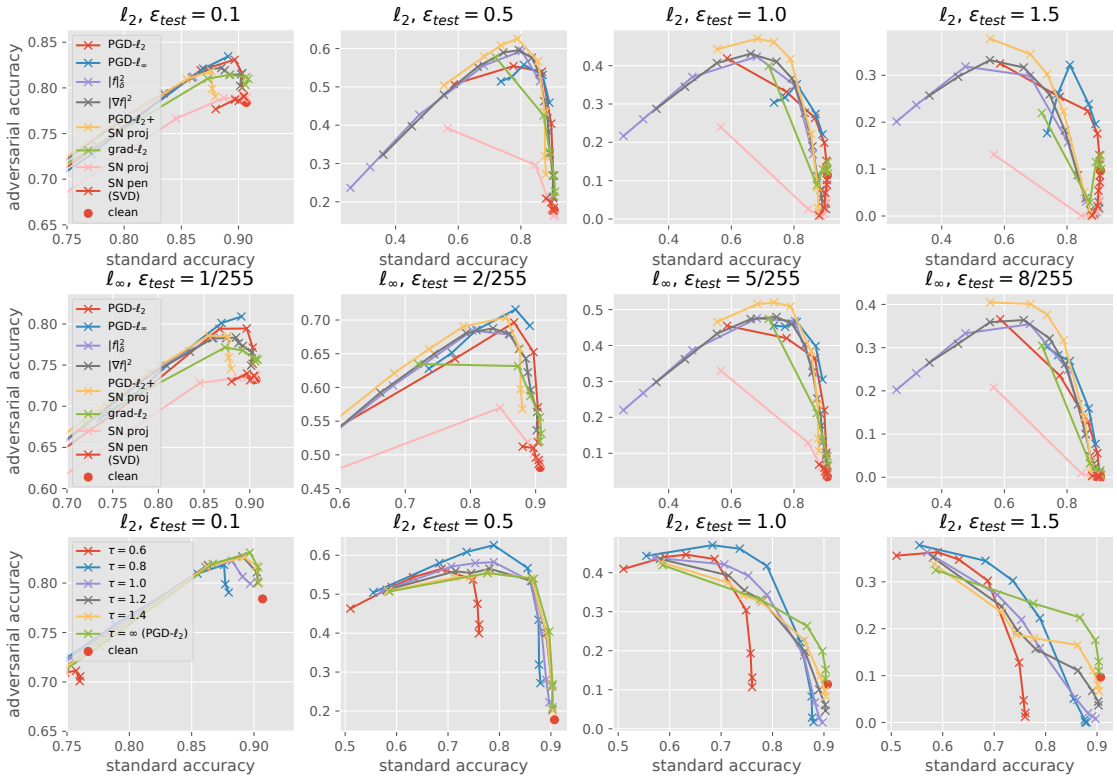


Figure 3.A.1: Robustness trade-off curves of different regularization methods for VGG11 on CIFAR10 (extended version of Figure 3.1). The plots show test accuracy vs adversarial test accuracy for ℓ_2-bounded (top/bottom) or ℓ_∞-bounded (middle), 100-step PGD adversaries with a fixed ϵ_{test}. Different points on a curve correspond to training with different regularization strengths. The regularization increases monotonically along a given curve, and the leftmost points correspond to the strongest regularization. The bottom plots consider PGD-ℓ_2 + SN projection, with different fixed values of the constraint radius τ, for varying ϵ in PGD.

Table 3.A.1: Regularization on CIFAR10 with 1 000 or 5 000 examples for VGG-11 and ResNet-18. Extended version of Table 3.1. Each entry shows the test accuracy with/without data augmentation when all hyper-parameters are optimized on a validation set of size 10 000 (a) or 1 000 (b), and for the epoch with highest validation accuracy, evaluating every 10 epochs (similar to early stopping).

(a) 10k examples in validation set

Method	1k VGG-11	1k ResNet-18	5k VGG-11	5k ResNet-18
No weight decay	50.70 / 43.75	45.23 / 37.12	72.49 / 58.35	72.72 / 54.12
Weight decay	51.32 / 43.95	44.85 / 37.09	72.80 / 58.56	73.06 / 53.33
SN penalty (PI)	54.64 / 45.06	47.01 / 39.63	74.03 / 62.45	74.79 / 54.04
SN penalty (SVD)	53.44 / 46.06	47.26 / 37.94	74.53 / 62.93	75.59 / 54.98
SN projection	54.14 / **46.70**	47.12 / 37.28	75.14 / 63.81	76.23 / 55.60
VAT	50.88 / 43.36	47.47 / 42.82	72.91 / 58.78	71.56 / 55.93
PGD-ℓ_2	51.25 / 44.40	45.80 / 41.87	73.18 / 58.98	72.53 / 55.92
PGD-ℓ_∞	51.17 / 43.07	45.31 / 39.66	73.05 / 57.82	72.75 / 55.14
grad-ℓ_2	**55.19** / 43.88	**49.30** / **44.65**	**75.38** / 59.20	75.22 / 55.36
grad-ℓ_1	54.88 / 44.74	**49.06** / 42.63	**75.25** / 59.39	74.48 / 56.19
$\|f\|_\delta^2$ penalty	51.41 / 45.07	48.73 / 43.72	72.98 / 61.45	72.78 / 56.50
$\|\nabla f\|^2$ penalty	54.80 / 46.37	**48.99** / **44.97**	73.90 / 60.17	73.83 / **57.92**
PGD-ℓ_2 + SN proj	54.19 / **46.66**	47.47 / 41.25	74.61 / **64.50**	**77.19** / 57.43
grad-ℓ_2 + SN proj	**55.32** / **46.88**	48.73 / 42.78	75.11 / 63.54	**77.73** / 57.09
$\|f\|_\delta^2$ + SN proj	54.02 / **46.72**	48.12 / 43.56	74.55 / **64.33**	75.64 / **59.03**
$\|\nabla f\|^2$ + SN proj	**55.24** / **46.80**	**49.06** / **44.92**	72.31 / 63.74	72.24 / 57.56

(b) 1k examples in validation set

Method	1k VGG-11	1k ResNet-18	5k VGG-11	5k ResNet-18
No weight decay	51.32 / 43.42	45.00 / 37.00	72.64 / 57.88	72.71 / 53.80
Weight decay	51.04 / 43.42	44.66 / 36.77	72.68 / 57.59	72.25 / 54.16
SN penalty (PI)	54.60 / 44.20	46.39 / 38.86	72.99 / 62.49	74.72 / 53.65
SN penalty (SVD)	53.76 / 44.79	47.31 / 37.92	74.05 / 63.34	75.73 / 54.65
SN projection	52.86 / **46.49**	47.05 / 37.28	74.18 / **63.70**	75.91 / 54.43
VAT	50.90 / 43.99	47.35 / 42.91	72.95 / 57.64	71.91 / 55.22
PGD-ℓ_2	50.95 / 43.26	45.77 / 41.71	72.71 / 57.68	72.87 / 54.17
PGD-ℓ_∞	51.16 / 43.16	45.67 / 39.77	73.64 / 58.02	72.99 / 53.95
grad-ℓ_2	**55.40** / 43.57	47.86 / **44.65**	**75.44** / 58.33	74.83 / 55.43
grad-ℓ_1	54.53 / 43.04	**48.75** / 42.21	**75.28** / 58.19	74.28 / 54.02
$\|f\|_M^2$ penalty	51.00 / 44.67	48.57 / 44.30	72.76 / 60.55	72.75 / 56.49
$\|\nabla f\|^2$ penalty	54.68 / 46.10	48.53 / **45.21**	73.83 / 60.36	73.30 / **57.46**
PGD-ℓ_2 + SN proj	53.85 / **46.79**	46.48 / 40.95	74.79 / 63.37	**76.28** / **57.43**
grad-ℓ_2 + SN proj	**55.28** / 45.11	48.42 / 41.93	75.17 / 63.45	**77.24** / 56.18
$\|f\|_M^2$ + SN proj	54.00 / 45.14	47.12 / 41.86	74.54 / **63.94**	75.25 / **57.94**
$\|\nabla f\|^2$ + SN proj	**55.21** / 45.68	**49.03** / 43.58	71.92 / 63.47	71.83 / 56.06

Table 3.A.2: Paired t-tests comparing pairs of methods, on 10 different random choices of subsets of CIFAR10. Each cell shows the p-value of the corresponding test, both with (left) and without (right) data augmentation. We only show p-values smaller than 0.05. Hyperparameters are fixed to the ones obtained for the results in Table 3.1 (selected on a different choice of subset), except for the learning rate which is tuned on a separate validation set for each choice of subset.

Test	1k VGG-11		1k ResNet-18		5k VGG-11		5k ResNet-18	
SN projection $\succ$ Weight decay	1e-04	1e-03	-	-	3e-06	1e-08	9e-07	4e-04
grad-ℓ_2 $\succ$ Weight decay	4e-09	-	2e-04	5e-05	7e-08	1e-04	5e-06	-
$\|\nabla f\|^2 \succ$ Weight decay	1e-08	2e-07	1e-05	3e-07	3e-04	5e-07	7e-03	1e-06
$\|\nabla f\|^2 \succ$ grad-ℓ_2	-	3e-08	2e-02	2e-06	-	6e-05	-	4e-05
grad-$\ell_2 \succ \|\nabla f\|^2$	2e-02	-	-	-	2e-05	-	7e-04	-
grad-ℓ_2 + SN proj $\succ$ grad-ℓ_2	-	9e-03	-	-	-	5e-07	9e-06	2e-04
$\|\nabla f\|^2$ + SN proj $\succ \|\nabla f\|^2$	-	-	-	1e-02	-	2e-06	-	-

Table 3.A.3: List of hyper-parameters used for each method on CIFAR10. For each method, we additionally consider a learning rate parameter in $[0.003; 0.01; 0.03; 0.1]$. For combined penalties, the sets of hyperparameters are listed in the same order as in the first column (*i.e.*, the choices of constraint radius are given last).

Method	Parameter grid
No weight decay	-
Weight decay	$[0; 0.0001; 0.0002; 0.0004; 0.0008; 0.001; 0.002]$
SN penalty (PI)	$[0.001; 0.003; 0.01; 0.03; 0.1; 0.3]$
SN penalty (SVD)	$[0.001; 0.003; 0.01; 0.03; 0.1; 0.3]$
SN projection	$[0.5; 0.6; 0.8; 1.0; 1.2; 1.4]$
$\|f\|_\delta^2$ penalty	$[0.001; 0.003; 0.01; 0.03; 0.1]$
$\|\nabla f\|^2$ penalty	$[0.00003; 0.0001; 0.0003; 0.001; 0.003; 0.01; 0.03]$
VAT	$[0.1; 0.3; 1.0; 3.0]$
PGD-ℓ_2	$[0.003; 0.01; 0.03; 0.1; 0.3; 1.0]$
PGD-ℓ_∞	$[0.001; 0.003; 0.01; 0.03; 0.1; 0.3]$
grad-ℓ_1	$[0.0001; 0.0003; 0.001; 0.003; 0.01; 0.03]$
grad-ℓ_2	$[0.001; 0.003; 0.01; 0.03; 0.1; 0.3; 1.0; 3.0]$
PGD-ℓ_2 + SN projection	$[0.003; 0.01; 0.03; 0.1] \times [0.6; 1.0; 1.4]$
grad-ℓ_2 + SN projection	$[0.003; 0.01; 0.03; 0.1] \times [0.6; 1.0; 1.4]$
$\|f\|_\delta^2$ + SN projection	$[0.003; 0.01; 0.03] \times [0.6; 1.0; 1.4]$
$\|\nabla f\|^2$ + SN projection	$[0.001; 0.01; 0.1] \times [0.6; 1.0; 1.4]$

Table 3.A.4: Test accuracies on subsets of MNIST using deformations from Infinite MNIST. Extended version of Table 3.2. $(*)$ indicates that random deformations were included as training examples (*i.e.*, data augmentation), while $\|f\|_\tau^2$ and $\|D_\tau f\|^2$ use them as part of the regularization penalty. As in Table 3.A.1, we show results obtained using a validation set of size 10 000 (a) and 1 000 (b).

(a) 10k examples in validation set

Method	300 VGG	1k VGG
Weight decay	89.32	94.08
Weight decay $(*)$	92.41	95.64
SN projection	90.69	95.01
SN projection $(*)$	92.17	95.88
grad-ℓ_2	93.63	96.67
grad-ℓ_2 $(*)$	95.05	97.48
$\|f\|_\delta^2$ penalty	94.17	96.99
$\|f\|_\delta^2$ penalty $(*)$	94.86	97.40
$\|\nabla f\|^2$ penalty	94.08	96.82
$\|\nabla f\|^2$ penalty $(*)$	94.80	97.29
$\|D_\tau f\|^2$ penalty	94.18	96.98
$\|D_\tau f\|^2$ penalty $(*)$	94.91	97.29
$\|f\|_\tau^2$ penalty	94.42	97.13
$\|f\|_\tau^2$ penalty $(*)$	94.83	97.25
$\|f\|_\tau^2 + \|\nabla f\|^2$	94.75	97.40
$\|f\|_\tau^2 + \|\nabla f\|^2$ $(*)$	95.14	97.44
$\|f\|_\tau^2 + \|f\|_\delta^2$	95.23	**97.66**
$\|f\|_\tau^2 + \|f\|_\delta^2$ $(*)$	**95.53**	**97.56**
grad-ℓ_2 + SN proj	93.89	96.85
grad-ℓ_2 + SN proj $(*)$	95.15	**97.80**
$\|f\|_\delta^2$ + SN proj	93.97	96.89
$\|f\|_\delta^2$ + SN proj $(*)$	94.78	97.38
$\|f\|_\tau^2 + \|\nabla f\|^2$ + SN proj	95.09	97.42
$\|f\|_\tau^2 + \|\nabla f\|^2$ + SN proj $(*)$	95.03	97.27
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj	95.20	**97.60**
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj $(*)$	**95.40**	**97.77**

(b) 1k examples in validation set

Method	300 VGG	1k VGG
Weight decay	89.32	93.34
Weight decay $(*)$	91.91	95.73
SN projection	90.60	94.83
SN projection $(*)$	92.01	95.91
grad-ℓ_2	92.92	96.42
grad-ℓ_2 $(*)$	**94.69**	**97.48**
$\|f\|_M^2$ penalty	93.44	96.98
$\|f\|_M^2$ penalty $(*)$	94.57	97.14
$\|\nabla f\|^2$ penalty	94.08	96.77
$\|\nabla f\|^2$ penalty $(*)$	94.50	97.15
$\|D_\tau f\|^2$ penalty	94.03	97.16
$\|D_\tau f\|^2$ penalty $(*)$	94.15	96.64
$\|f\|_\tau^2$ penalty	93.53	97.13
$\|f\|_\tau^2$ penalty $(*)$	**94.79**	97.26
$\|f\|_\tau^2 + \|\nabla f\|^2$	**94.75**	97.21
$\|f\|_\tau^2 + \|\nabla f\|^2$ $(*)$	94.43	**97.42**
$\|f\|_\tau^2 + \|f\|_M^2$	**95.15**	97.27
$\|f\|_\tau^2 + \|f\|_M^2$ $(*)$	**95.20**	**97.49**
grad-ℓ_2 + SN proj	93.44	96.81
grad-ℓ_2 + SN proj $(*)$	94.05	**97.60**
$\|f\|_M^2$ + SN proj	93.97	96.61
$\|f\|_M^2$ + SN proj $(*)$	**94.69**	97.33
$\|f\|_\tau^2 + \|\nabla f\|^2$ + SN proj	**94.75**	97.16
$\|f\|_\tau^2 + \|\nabla f\|^2$ + SN proj $(*)$	**94.74**	97.22
$\|f\|_\tau^2 + \|f\|_M^2$ + SN proj	**94.78**	**97.49**
$\|f\|_\tau^2 + \|f\|_M^2$ + SN proj $(*)$	**95.17**	**97.64**

Table 3.A.5: Paired t-tests comparing pairs of methods, on 10 different random choices of subsets of MNIST. Each cell shows the p-value of the corresponding test. We only show p-values smaller than 0.05. Hyperparameters are fixed to the ones obtained for the results in Table 3.2 (selected on a different choice of subset), except for the learning rate which is tuned on a separate validation set for each choice of subset.

Test	300 VGG	1k VGG
grad-ℓ_2 $(*)$ $\succ$ Weight decay $(*)$	-	3e-11
$\|f\|_\tau^2$ penalty $\succ$ Weight decay $(*)$	2e-08	2e-10
$\|f\|_\tau^2 + \|f\|_\delta^2$ $\succ$ Weight decay $(*)$	1e-08	2e-10
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj $(*)$ $\succ$ grad-ℓ_2 $(*)$	-	1e-02
grad-ℓ_2 $(*)$ $\succ$ $\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj $(*)$	-	-
$\|f\|_\tau^2 + \|f\|_\delta^2$ $\succ$ $\|f\|_\delta^2$ penalty	1e-07	6e-09
$\|f\|_\tau^2 + \|f\|_\delta^2$ $\succ$ $\|f\|_\tau^2$ penalty	2e-06	6e-07
$\|f\|_\tau^2 + \|f\|_\delta^2$ $(*)$ $\succ$ $\|f\|_\tau^2 + \|f\|_\delta^2$	2e-03	-
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj $(*)$ $\succ$ $\|f\|_\tau^2 + \|f\|_\delta^2$	2e-03	2e-04

Table 3.A.6: List of hyper-parameters used for each method on Infinite MNIST. For each method, we additionally consider a learning rate parameter in $[0.005; 0.05; 0.5]$. For combined penalties, the sets of hyperparameters are listed in the same order as in the first column (*e.g.*, the choices of constraint radius are given last).

Method	Grid
Weight decay	[0; 0.00001; 0.00003; 0.0001; 0.0003; 0.001; 0.003; 0.01; 0.03; 0.1]
SN projection	[1.0; 1.2; 1.4; 1.6; 1.8]
grad-ℓ_2	[0.1; 0.3; 1.0; 3.0; 10.0]
$\|f\|_\delta^2$ penalty	[0.1; 0.3; 1.0; 3.0]
$\|\nabla f\|^2$ penalty	[0.0003; 0.001; 0.003; 0.01; 0.03; 0.1; 0.3]
$\|D_\tau f\|^2$ penalty	[0.003; 0.01; 0.03; 0.1; 0.3]
$\|f\|_\tau^2$ penalty	[0.03; 0.1; 0.3; 1.0; 3.0]
$\|f\|_\tau^2 + \|\nabla f\|^2$	[0.03; 0.1; 0.3; 1.0] $\times$ [0.003; 0.01; 0.03; 0.1]
$\|f\|_\tau^2 + \|f\|_\delta^2$	[0.1; 0.3; 1.0] $\times$ [0.03; 0.1]
grad-ℓ_2 + SN proj	[0.3; 1.0; 3.0; 10.0; 30.0] $\times$ [1.2; 1.6; 2.0]
$\|f\|_\delta^2$ + SN proj	[0.03; 0.1] $\times$ [1.2; 1.6; 2.0]
$\|f\|_\tau^2 + \|\nabla f\|^2$ + SN proj	[0.03; 0.1; 0.3] $\times$ [0.01; 0.03; 0.1] $\times$ [1.2; 1.6; 2.0]
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj	[0.1; 0.3; 1.0] $\times$ [0.03; 0.1] $\times$ [1.2; 1.6; 2.0]

Table 3.A.7: Paired t-tests comparing pairs of methods on the 51 test datasets from the set of protein homology detection tasks. Each cell shows the p-value of the corresponding test. We only show p-values smaller than 0.05. We use the same hyperparameters as the ones obtained in the results of Table 3.3.

Test	No DA	DA
SN proj $\succ$ Weight decay	1e-05	4e-05
grad-$\ell_2 \succ$ Weight decay	5e-05	5e-02
$\|f\|_\delta^2 \succ$ Weight decay	5e-06	3e-05
$\|\nabla f\|^2 \succ$ Weight decay	9e-06	3e-03
$\|f\|_\delta^2 \succ$ grad-ℓ_2	-	4e-03
$\|\nabla f\|^2 \succ$ grad-ℓ_2	-	-
grad-ℓ_2 + SN proj $\succ$ grad-ℓ_2	-	1e-03
$\|f\|_\delta^2$ + SN proj $\succ \|f\|_\delta^2$	3e-03	5e-02
$\|\nabla f\|^2$ + SN proj $\succ \|\nabla f\|^2$	-	-
$\|f\|_\delta^2$ + SN proj $\succ \|\nabla f\|^2$ + SN proj	8e-05	-

Table 3.A.8: List of hyper-parameters used for each method on protein homology detection datasets. For combined penalties, the hyperparameters are the cross-products of each individual method.

Method	Parameter grid
No weight decay	–
Weight decay	$[0; 0.01; 0.001; 0.0001; 0.00001]$
SN proj	$[10; 1.0; 0.1]$
PGD-ℓ_2	$[100.0; 10.0; 1.0; 0.1]$
grad-ℓ_2	$[100.0; 10.0; 1.0; 0.1; 0.01, 0.001]$
$\|f\|_\delta^2$	$[10.0; 1.0; 0.1]$
$\|\nabla f\|^2$	$[10.0; 1.0; 0.1; 0.01; 0.001; 0.0001]$

Chapter 4

Links with Optimization: Inductive Bias of Neural Tangent Kernels

State-of-the-art neural networks are heavily over-parameterized, making the optimization algorithm a crucial ingredient for learning predictive models with good generalization properties. A recent line of work has shown that in a certain over-parameterized regime, the learning dynamics of gradient descent are governed by a certain kernel obtained at initialization, called the *neural tangent kernel*. We study the inductive bias of learning in such a regime by analyzing this kernel and the corresponding function space (RKHS). In particular, we study smoothness, approximation, and stability properties of functions with finite norm, including stability to image deformations in the case of convolutional networks.

This chapter is based on the following paper:

A. Bietti and J. Mairal. On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019b

4.1 Introduction

The large number of parameters in state-of-the-art deep neural networks makes them very expressive, with the ability to approximate large classes of functions (Hornik et al., 1989; Pinkus, 1999). Since many networks can potentially fit a given dataset, the optimization method, typically a variant of gradient descent, plays a crucial role in selecting a model that generalizes well (Neyshabur et al., 2015b).

A recent line of work (Allen-Zhu et al., 2019b; Chizat et al., 2019; Du et al., 2019a,b; Jacot et al., 2018) has shown that when training deep networks in a certain over-parameterized regime, the dynamics of gradient descent behave like those of a linear model on (non-linear) features determined at initialization. In the over-parameterization limit, these features correspond to a kernel known as the *neural tangent kernel*. In particular, in the case of a regression loss, the obtained model behaves similarly to a minimum norm kernel least squares solution, suggesting that this kernel may play a key role in determining the inductive bias of the learning procedure and its generalization proper-

ties. While it is still not clear if this regime is at play in state-of-the-art deep networks, there is some evidence that this phenomenon of "lazy training" (Chizat et al., 2019), where weights only move very slightly during training, may be relevant for early stages of training and for the outmost layers of deep networks (Lee et al., 2019; Zhang et al., 2019), motivating a better understanding of its properties.

In this paper, we study the inductive bias of this regime by studying properties of functions in the space associated with the neural tangent kernel for a given architecture (that is, the reproducing kernel Hilbert space, or RKHS). Such kernels can be defined recursively using certain choices of dot-product kernels at each layer that depend on the activation function. For the convolutional case with rectified linear unit (ReLU) activations and arbitrary patches and linear pooling operations, we show that the NTK can be expressed through kernel feature maps defined in a tree-structured hierarchy.

We study smoothness and stability properties of the kernel mapping for two-layer networks and CNNs, which control the variations of functions in the RKHS. In particular, a useful inductive bias when dealing with natural signals such as images is stability of the output to deformations of the input, such as translations or small rotations. A precise notion of stability to deformations was proposed by Mallat (2012), and we study it in Chapter 2 in the context of CNN architectures, showing the benefits of different architectural choices such as small patch sizes. In contrast to the kernels studied in Chapter 2, which for instance cover the limiting kernels that arise from training only the last layer of a ReLU CNN, we find that the obtained NTK kernel mappings for the ReLU activation lack a desired Lipschitz property which is needed for stability to deformations in the sense of Mallat (2012). Instead, we show that a weaker smoothness property similar to Hölder smoothness holds, and this allows us to show that the kernel mapping is stable to deformations, albeit with a different guarantee.

In order to balance our observations on smoothness, we also consider approximation properties for the NTK of two-layer ReLU networks, by characterizing the RKHS using a Mercer decomposition of the kernel in the basis of spherical harmonics (Bach, 2017a; Schölkopf and Smola, 2001; Smola et al., 2001). In particular, we study the decay of eigenvalues for this decomposition, which is then related to the regularity of functions in the space, and provides rates of approximation for Lipschitz functions (Bach, 2017a). We find that the full NTK has better approximation properties compared to other function classes typically defined for ReLU activations (Bach, 2017a; Cho and Saul, 2009; Daniely et al., 2016), which arise for instance when only training the weights in the last layer, or when considering Gaussian process limits of ReLU networks (*e.g.*, Garriga-Alonso et al., 2019; Lee et al., 2018; Matthews et al., 2018; Novak et al., 2019).

Contributions. Our main contributions can be summarized as follows:

- We provide a derivation of the NTK for convolutional networks with generic linear operators for patch extraction and pooling, and express the corresponding kernel feature map hierarchically using these operators.
- We study smoothness properties of the kernel mapping for ReLU networks, showing that it is not Lipschitz but satisfies a weaker Hölder smoothness property. For CNNs, we then provide a guarantee on deformation stability.
- We characterize the RKHS of the NTK for two-layer ReLU networks by providing a spectral decomposition of the kernel and studying its spectral decay. This leads

to improved approximation properties compared to other function classes based on ReLU.

Related work. Neural tangent kernels were introduced by Jacot et al. (2018), and similar ideas were used to obtain more quantitative guarantees on the global convergence of gradient descent for over-parameterized neural networks (Allen-Zhu et al., 2019b; Arora et al., 2019a; Chizat et al., 2019; Du et al., 2019a,b; Li and Liang, 2018; Zou et al., 2019). Arora et al. (2019a); Du et al. (2019a); Yang (2019) also derive NTKs for convolutional networks, but focus on simpler architectures. Kernel methods for deep neural networks were studied for instance by Cho and Saul (2009); Daniely et al. (2016); Mairal (2016). Stability to deformations was originally introduced in the context of the scattering representation (Bruna and Mallat, 2013; Mallat, 2012), and this thesis later extended such a study to neural networks through kernel methods, see Chapter 2. The inductive bias of optimization in neural network learning was considered, *e.g.*, by Allen-Zhu et al. (2019a); Arora et al. (2019b); Cao and Gu (2019); Neyshabur et al. (2015b); Soudry et al. (2018). Bach (2017a); Ghorbani et al. (2019); Savarese et al. (2019); Williams et al. (2019) study function spaces corresponding to two-layer ReLU networks. In particular, Ghorbani et al. (2019) also analyzes properties of the NTK, but studies a specific high-dimensional limit for generic activations, while we focus on ReLU networks, studying the corresponding eigenvalue decays in finite dimension. We note that Basri et al. (2019); Xie et al. (2017); Yang and Salman (2019) also study spectral properties of similar kernels in different contexts.

4.2 Neural Tangent Kernels

In this section, we provide some background on "lazy training" and neural tangent kernels (NTKs), and introduce the kernels that we study in this paper. In particular, we derive the NTK for generic convolutional architectures on ℓ^2 signals. For simplicity of exposition, we consider scalar-valued functions, noting that the kernels may be extended to the vector-valued case, as done, *e.g.*, by Jacot et al. (2018).

4.2.1 Lazy training and neural tangent kernels

Multiple recent works studying global convergence of gradient descent in neural networks (*e.g.*, Allen-Zhu et al., 2019b; Du et al., 2019a,b; Jacot et al., 2018; Li and Liang, 2018; Zou et al., 2019) show that when a network is sufficiently over-parameterized, weights remain close to initialization during training. The model is then well approximated by its linearization around initialization. For a neural network $f(x;\theta)$ with parameters θ and initialization θ_0, we then have:[1]

$$f(x;\theta) \approx f(x;\theta_0) + \langle \theta - \theta_0, \nabla_\theta f(x;\theta_0) \rangle. \tag{4.1}$$

This regime where weights barely move has also been referred to as "lazy training" Chizat et al. (2019), in contrast to other situations such as the "mean-field" regime (*e.g.*, Chizat and Bach (2018); Mei et al. (2018, 2019)), where weights move according to non-linear

[1] While we use gradients in our notations, we note that weak differentiability (*e.g.*, with ReLU activations) is sufficient when studying the limiting NTK (Jacot et al., 2018).

dynamics. Yet, with sufficient over-parameterization, the (non-linear) features $x \mapsto \nabla_\theta f(x;\theta_0)$ of the linearized model (4.1) become expressive enough to be able to perfectly fit the training data, by approximating a kernel method.

Neural Tangent Kernel (NTK). When the width of the network tends to infinity, assuming an appropriate initialization on weights, the features of the linearized model tend to a limiting kernel K, called *neural tangent kernel* (Jacot et al., 2018):

$$\langle \nabla_\theta f(x;\theta_0), \nabla_\theta f(x',\theta_0)\rangle \to K(x,x'). \tag{4.2}$$

In this limit and under some assumptions, one can show that the weights move very slightly and the kernel remains fixed during training, and that gradient descent will then lead to the minimum norm kernel least-squares fit of the training set in the case of the ℓ_2 loss (Jacot et al., 2018). Similar interpolating solutions have been found to perform well for generalization, both in practice (Belkin et al., 2018b) and in theory (Bartlett et al., 2019; Liang and Rakhlin, 2019). When the number of neurons is large but finite, one can often show that the kernel only deviates slightly from the limiting NTK, at initialization and throughout training, thus allowing convergence as long as the initial kernel matrix is non-degenerate (Arora et al., 2019a; Chizat et al., 2019; Du et al., 2019a,b).

NTK for two-layer ReLU networks. Consider a two layer network of the form $f(x;\theta) = \sqrt{\frac{2}{m}} \sum_{j=1}^m v_j \sigma(w_j^\top x)$, where $\sigma(u) = (u)_+ = \max(0,u)$ is the ReLU activation, $x \in \mathbb{R}^p$, and $\theta = (w_1^\top, \dots, w_m^\top, v^\top)$ are parameters with values initialized as $\mathcal{N}(0,1)$. Practitioners often include the factor $\sqrt{2/m}$ in the variance of the initialization of v_j, but we treat it as a scaling factor following Du et al. (2019a,b); Jacot et al. (2018), noting that this leads to the same predictions. The factor 2 is simply a normalization constant specific to the ReLU activation and commonly used by practitioners, which avoids vanishing or exploding behavior for deep networks. The corresponding NTK is then given by (Chizat et al., 2019; Du et al., 2019b):

$$\begin{aligned} K(x,x') &= 2(x^\top x')\, \mathbb{E}_{w\sim\mathcal{N}(0,I)}[1\{w^\top x \geq 0\} 1\{w^\top x' \geq 0\}] + 2\, \mathbb{E}_{w\sim\mathcal{N}(0,I)}[(w^\top x)_+ (w^\top x')_+] \\ &= \|x\|\|x'\| \kappa\left(\frac{\langle x, x'\rangle}{\|x\|\|x'\|}\right), \end{aligned} \tag{4.3}$$

where

$$\kappa(u) := u\kappa_0(u) + \kappa_1(u) \tag{4.4}$$

$$\kappa_0(u) = \frac{1}{\pi}\left(\pi - \arccos(u)\right), \qquad \kappa_1(u) = \frac{1}{\pi}\left(u\cdot(\pi - \arccos(u)) + \sqrt{1-u^2}\right). \tag{4.5}$$

The expressions for κ_0 and κ_1 follow from standard calculations for arc-cosine kernels of degree 0 and 1 (see Cho and Saul, 2009). Note that in this two-layer case, the non-linear features obtained for finite neurons correspond to a random features kernel (Rahimi and Recht, 2007), which is known to approximate the full kernel relatively well even with a moderate amount of neurons (Bach, 2017b; Rahimi and Recht, 2007; Rudi and Rosasco, 2017). One can also extend the derivation to other activation functions, which may lead to explicit expressions for the kernel in some cases (Daniely et al., 2016).

NTK for fully-connected deep ReLU networks. We define a fully-connected neural network by $f(x;\theta) = \sqrt{\frac{2}{m_n}}\langle w^{n+1}, a^n\rangle$, with $a^1 = \sigma(W^1 x)$, and

$$a^k = \sigma\left(\sqrt{\frac{2}{m_{k-1}}} W^k a^{k-1}\right), \quad k = 2, \dots, n,$$

where $W^k \in \mathbb{R}^{m_k \times m_{k-1}}$ and $w^{n+1} \in \mathbb{R}^{m_n}$ are initialized with i.i.d. $\mathcal{N}(0,1)$ entries, and $\sigma(u) = (u)_+$ is the ReLU activation and is applied element-wise. Following Jacot et al. (2018), the corresponding NTK is defined recursively by $K(x,x') = K_n(x,x')$ with $K_0(x,x') = \Sigma_0(x,x') = x^\top x'$, and for $k \geq 1$,

$$\begin{aligned}
\Sigma_k(x,x') &= 2\,\mathbb{E}_{(u,v)\sim\mathcal{N}(0,B_k)}[\sigma(u)\sigma(v)] \\
K_k(x,x') &= \Sigma_k(x,x') + 2K_{k-1}(x,x')\,\mathbb{E}_{(u,v)\sim\mathcal{N}(0,B_k)}[\sigma'(u)\sigma'(v)],
\end{aligned}$$

where $B_k = \begin{pmatrix} \Sigma_{k-1}(x,x) & \Sigma_{k-1}(x,x') \\ \Sigma_{k-1}(x,x') & \Sigma_{k-1}(x',x') \end{pmatrix}$. Using a change of variables and definitions of arc-cosine kernels of degrees 0 and 1 (Cho and Saul, 2009), it is easy to show that

$$2\,\mathbb{E}_{(u,v)\sim\mathcal{N}(0,B_k)}[\sigma(u)\sigma(v)] = \sqrt{\Sigma_{k-1}(x,x)\Sigma_{k-1}(x',x')}\,\kappa_1\left(\frac{\Sigma_{k-1}(x,x')}{\sqrt{\Sigma_{k-1}(x,x)\Sigma_{k-1}(x',x')}}\right) \tag{4.6}$$

$$2\,\mathbb{E}_{(u,v)\sim\mathcal{N}(0,B_k)}[\sigma'(u)\sigma'(v)] = \kappa_0\left(\frac{\Sigma_{k-1}(x,x')}{\sqrt{\Sigma_{k-1}(x,x)\Sigma_{k-1}(x',x')}}\right), \tag{4.7}$$

where κ_0 and κ_1 are defined in (4.5).

Feature maps construction. We now provide a reformulation of the previous kernel in terms of explicit feature maps, which provides a representation of the data and makes our study of stability in Section 2.3 more convenient. For a given input Hilbert space $\mathcal{H}$, we denote by $\varphi_{\mathcal{H},1} : \mathcal{H} \to \mathcal{H}_1$ the kernel mapping into the RKHS $\mathcal{H}_1$ for the kernel $(z,z') \in \mathcal{H}^2 \mapsto \|z\|\|z'\|\kappa_1(\langle z,z'\rangle/\|z\|\|z'\|)$, and by $\varphi_{\mathcal{H},0} : \mathcal{H} \to \mathcal{H}_0$ the kernel mapping into the RKHS $\mathcal{H}_0$ for the kernel $(z,z') \in \mathcal{H}^2 \mapsto \kappa_0(\langle z,z'\rangle/\|z\|\|z'\|)$. We will abuse notation and hide the input space, simply writing φ_1 and φ_0.

Lemma 4.1 (NTK feature map for fully-connected network)**.** *The NTK for the fully-connected network can be defined as $K(x,x') = \langle \Phi_n(x), \Phi_n(x')\rangle$, with $\Phi_0(x) = \Psi_0(x) = x$ and for $k \geq 1$,*

$$\begin{aligned}
\Psi_k(x) &= \varphi_1(\Psi_{k-1}(x)) \\
\Phi_k(x) &= \begin{pmatrix} \varphi_0(\Psi_{k-1}(x)) \otimes \Phi_{k-1}(x) \\ \varphi_1(\Psi_{k-1}(x)) \end{pmatrix},
\end{aligned}$$

where $\otimes$ is the tensor product.

4.2.2 Neural tangent kernel for convolutional networks

In this section we study NTKs for convolutional networks (CNNs) on signals, focusing on the ReLU activation. We consider signals in $\ell^2(\mathbb{Z}^d, \mathbb{R}^{m_0})$, that is, signals $x[u]$ with $u \in \mathbb{Z}^d$ denoting the location, $x[u] \in \mathbb{R}^{m_0}$, and $\sum_{u\in\mathbb{Z}^d} \|x[u]\|^2 < \infty$ (for instance, $d = 2$ and $m_0 = 3$ for RGB images). The infinite support allows us to avoid dealing with boundary conditions when considering deformations and pooling. The precise study of ℓ^2 membership is deferred to Section 2.3.

Patch extraction and pooling operators P^k and A^k. Following Chapter 2, we define two linear operators P^k and A^k on $\ell^2(\mathbb{Z}^d)$ for extracting patches and performing (linear) pooling at layer k, respectively. For an $\mathcal{H}$-valued signal $x[u]$, P^k is defined by $P^k x[u] = |S_k|^{-1/2}(x[u+v])_{v\in S_k} \in \mathcal{H}^{|S_k|}$, where S_k is a finite subset of $\mathbb{Z}^d$ defining the patch shape (*e.g.*, a 3x3 box). Pooling is defined as a convolution with a linear filter $h_k[u]$, *e.g.*, a Gaussian filter at scale σ_k as in Chapter 2, that is, $A^k x[u] = \sum_{v\in\mathbb{Z}^d} h_k[u-v]x[v]$. In this discrete setting, we can easily include a downsampling operation with factor s_k by changing the definition of A^k to $A^k x[u] = \sum_{v\in\mathbb{Z}^d} h_k[s_k u - v]x[v]$ (in particular, if h_k is a Dirac at 0, we obtain a CNN with "strided convolutions"). In fact, our NTK derivation supports general linear operators $A^k : \ell^2(\mathbb{Z}^d) \to \ell^2(\mathbb{Z}^d)$ on scalar signals.

For defining the NTK feature map, we also introduce the following non-linear pointwise operator M, given for two signals x, y, by

$$M(x,y)[u] = \begin{pmatrix} \varphi_0(x[u]) \otimes y[u] \\ \varphi_1(x[u]) \end{pmatrix}, \tag{4.8}$$

where $\varphi_{0/1}$ are kernel mappings of arc-cosine 0/1 kernels, as defined in Section 4.2.1.

CNN definition and NTK. We consider a network $f(x;\theta) = \sqrt{\frac{2}{m_n}}\langle w^{n+1}, a^n\rangle_{\ell^2}$, with

$$\begin{aligned} \tilde{a}^k[u] &= \begin{cases} W^1 P^1 x[u], & \text{if } k = 1, \\ \sqrt{\frac{2}{m_{k-1}}} W^k P^k a^{k-1}[u], & \text{if } k \in \{2,\dots,n\}, \end{cases} \\ a^k[u] &= A^k \sigma(\tilde{a}^k)[u], \quad k = 1,\dots,n, \end{aligned}$$

where $W^k \in \mathbb{R}^{m_k \times m_{k-1}|S_k|}$ and $w^n \in \ell^2(\mathbb{Z}^d, \mathbb{R}^{m_n})$ are initialized with $\mathcal{N}(0,1)$ entries, and $\sigma(\tilde{x}^k)$ denotes the signal with σ applied element-wise to $\tilde{x}^k$. We are now ready to state our result on the NTK for this model.

Proposition 4.2 (NTK feature map for CNN)**.** *The NTK for the above CNN, obtained when the number of feature maps $m_1,\dots,m_n \to \infty$ (sequentially), is given by $K(x,x') = \langle \Phi(x), \Phi(x')\rangle_{\ell^2(\mathbb{Z}^d)}$, with $\Phi(x)[u] = A^n M(x_n, y_n)[u]$, where y_n and x_n are defined recursively for a given input x by $y_1[u] = x_1[u] = P^1 x[u]$, and for $k \geq 2$,*

$$\begin{aligned} x_k[u] &= P^k A^{k-1} \varphi_1(x_{k-1})[u] \\ y_k[u] &= P^k A^{k-1} M(x_{k-1}, y_{k-1})[u], \end{aligned}$$

with the abuse of notation $\varphi_1(x)[u] = \varphi_1(x[u])$ for a signal x.

The proof is given in Appendix 4.B.2, where we also show that in the overparameterization limit, the pre-activations $\tilde{a}_i^k[u]$ tend to a Gaussian process with covariance $\Sigma^k(x,u;x',u') = \langle x_k[u], x_k'[u'] \rangle$ (this is related to recent papers by Garriga-Alonso et al. (2019); Novak et al. (2019) studying Gaussian process limits of Bayesian convolutional networks). The proof is by induction and relies on similar arguments to Jacot et al. (2018) for fully-connected networks, in addition to exploiting linearity of the operators P^k and A^k, as well as recursive feature maps for hierarchical kernels. The recent papers by Arora et al. (2019a); Yang (2019) also study NTKs for certain convolutional networks; in contrast to these works, our derivation considers general signals in $\ell^2(\mathbb{Z}^d)$, supports intermediate pooling or downsampling by changing A^k, and provides a more intuitive construction through kernel mappings and the operators P^k and A^k. Note that the feature maps x_k are defined independently from the y_k, and in fact correspond to more standard multi-layer deep kernel machines as in Chapter 2 (see also Cho and Saul, 2009; Daniely et al., 2016; Mairal, 2016) or covariance functions of certain deep Bayesian networks (Garriga-Alonso et al., 2019; Lee et al., 2018; Matthews et al., 2018; Novak et al., 2019). They can also be seen as the feature maps of the limiting kernel that arises when only training weights in the last layer and fixing other layers at initialization (see, *e.g.*, Daniely et al., 2016).

4.3 Two-Layer Networks

In this section, we study smoothness and approximation properties of the RKHS defined by neural tangent kernels for two-layer networks. For ReLU activations, we show that the NTK kernel mapping is not Lipschitz, but satisfies a weaker smoothness property. In Section 4.3.2, we characterize the RKHS for ReLU activations and study its approximation properties and benefits. Finally, we comment on the use of other activations in Section 4.3.3.

4.3.1 Smoothness of two-layer ReLU networks

Here we study the RKHS $\mathcal{H}$ of the NTK for two-layer ReLU networks, defined in (4.3), focusing on smoothness properties of the kernel mapping, denoted $\Phi(\cdot)$. Recall that smoothness of the kernel mapping guarantees smoothness of functions $f \in \mathcal{H}$, through the relation

$$|f(x) - f(y)| \leq \|f\|_{\mathcal{H}} \|\Phi(x) - \Phi(y)\|_{\mathcal{H}}. \tag{4.9}$$

We begin by showing that the kernel mapping for the NTK is not Lipschitz. This is in contrast to the kernel κ_1 in (4.5), obtained by fixing the weights in the first layer and training only the second layer weights (κ_1 is 1-Lipschitz by Lemma 2.1 in Chapter 2).

Proposition 4.3 (Non-Lipschitzness)**.** *The kernel mapping $\Phi(\cdot)$ of the two-layer NTK is not Lipschitz:*

$$\sup_{x,y} \frac{\|\Phi(x) - \Phi(y)\|_{\mathcal{H}}}{\|x - y\|} \to +\infty.$$

This is true even when looking only at points x, y on the sphere. It follows that the RKHS $\mathcal{H}$ contains unit-norm functions with arbitrarily large Lipschitz constant.

Note that the instability is due to φ_0, which comes from gradients w.r.t. first layer weigts. We now show that a weaker guarantee holds nevertheless, resembling 1/2-Hölder smoothness.

Proposition 4.4 (Smoothness for ReLU NTK)**.** *We have the following smoothness properties:*

1. *For x, y such that $\|x\| = \|y\| = 1$, the kernel mapping φ_0 satisfies $\|\varphi_0(x) - \varphi_0(y)\| \leq \sqrt{\|x - y\|}$.*
2. *For general non-zero x, y, we have $\|\varphi_0(x) - \varphi_0(y)\| \leq \sqrt{\frac{1}{\min(\|x\|,\|y\|)}\|x - y\|}$.*
3. *The kernel mapping Φ of the NTK then satisfies*

$$\|\Phi(x) - \Phi(y)\| \leq \sqrt{\min(\|x\|, \|y\|)\|x - y\|} + 2\|x - y\|.$$

We note that while such smoothness properties apply to the functions in the RKHS of the studied limiting kernels, the neural network functions obtained at finite width and their linearizations around initialization are not in the RKHS and thus may not preserve such smoothness properties, despite preserving good generalization properties, as in random feature models (Bach, 2017b; Rudi and Rosasco, 2017). This discrepancy may be a source of instability to adversarial perturbations.

4.3.2 Approximation properties for the two-layer ReLU NTK

In the previous section, we found that the NTK for two-layer ReLU networks yields weaker smoothness guarantees compared to the kernel κ_1 obtained when the first layer is fixed. We now show that the NTK has better approximation properties, by studying the RKHS through a spectral decomposition of the kernel and the decay of the corresponding eigenvalues. This highlights a tradeoff between smoothness and approximation.

Background on dot-product kernels and spherical harmonics. When restricting inputs to lie on the sphere in p dimensions, $\mathcal{X} = \mathbb{S}^{p-1} = \{x \in \mathbb{R}^p : \|x\| = 1\}$, the kernels we study take the form of dot-product kernels $K(x, y) = \kappa(\langle x, y\rangle)$, for which it is common to consider Mercer decompositions in the basis of spherical harmonics (*e.g.*, Smola et al., 2001; Vert and Mairal, 2017). More precisely, we may consider the integral operator T of κ defined on $L^2(\mathbb{S}^{p-1}, d\tau)$, where τ is the uniform measure on the sphere, given for $f \in L^2(\mathbb{S}^{p-1}, d\tau)$ by

$$Tf(x) = \int_{\mathbb{S}^{p-1}} \kappa(\langle x, y\rangle) f(y) d\tau(y).$$

This operator is then diagonalized in the basis of spherical harmonics, which consists of homogeneous polynomials forming an orthonormal basis of $L^2(\mathbb{S}^{p-1}, d\tau)$. For each degree k, which plays the role of an integer frequency as in Fourier, we have $N(p, k) = \frac{2k+p-2}{k}\binom{k+p-3}{p-2}$ spherical harmonics, denoted $Y_{k,j}(x)$ for $j = 1, \ldots, N(p, k)$. A useful tool for computing Fourier coefficients in this basis is the set of Legendre polynomials P_k, which form an orthogonal basis of $L^2([-1, 1], d\nu)$, with $d\nu(t) = (1 - t^2)^{(p-3)/2} dt$. We provide more background on spherical harmonics and Legendre polynomials in Appendix 4.A. Then, Mercer's theorem yields the following kernel decomposition (Schölkopf

and Smola, 2001; Smola et al., 2001):

$$\kappa(\langle x, y\rangle) = \sum_{k=0}^{\infty} \mu_k \sum_{j=1}^{N(p,k)} Y_{k,j}(x) Y_{k,j}(y) = \sum_{k=0}^{\infty} \mu_k N(p,k) P_k(\langle x, y\rangle), \tag{4.10}$$

where μ_k is obtained by computing Fourier coefficients of $\kappa(\langle x, \cdot\rangle)$ using the Funk-Hecke formula (see (4.22) in Appendix 4.A, ω_{p-1} is the surface of $\mathbb{S}^{p-1}$):

$$\mu_k = \frac{\omega_{p-2}}{\omega_{p-1}} \int_{-1}^{1} \kappa(t) P_k(t) (1-t^2)^{(p-3)/2} dt. \tag{4.11}$$

Note that we must have $\mu_k \geq 0$ for all k, since positive definiteness of k would be violated if this were not true. Then, the RKHS $\mathcal{H}$ consists of functions of the form

$$f(x) = \sum_{k=0}^{\infty} \sum_{j=1}^{N(p,k)} a_{k,j} Y_{k,j}(x), \tag{4.12}$$

$$\text{s.t.} \quad \|f\|_{\mathcal{H}}^2 = \sum_{k=0}^{\infty} \sum_{j=1}^{N(p,k)} \frac{a_{k,j}^2}{\mu_k} < \infty. \tag{4.13}$$

In particular, this requires that $a_{k,j} = 0$ for all j, for any k such that $\mu_k = 0$.

Relationship with differentiability. As in the case of Fourier series, there is a tight connection between regularity of function on the sphere, and decay of their Fourier coefficients in the basis of spherical harmonics. In particular, the decay of the eigenvalues μ_k is related to the regularity of functions in the RKHS. Following Bach (2017a, Appendix D.3), if f is s-times differentiable with derivatives bounded by η, then we have $\|(-\Delta)^{s/2} f\|_{L^2(\mathbb{S}^{p-1})} \leq \eta$, where Δ is the Laplace-Beltrami operator on the sphere (Atkinson and Han, 2012). Given a function f as in (4.12), we can use the fact that $Y_{k,j}$ is an eigenfunction of $-\Delta$ with eigenvalue $k(k+p-2)$ (see Atkinson and Han, 2012; Efthimiou and Frye, 2014) and write $a_{k,j} = a'_{k,j}/(k(k+p-2))^{s/2}$ for $k \geq 1$, where $a'_{k,j}$ are the Fourier coefficients of $(-\Delta)^{s/2} f$ and satisfy $\sum_{k,j} a'^2_{k,j} \leq \eta^2$. We then have

Lemma 4.5. *Assume f takes the form (4.12), with $a_{k,j} = 0$ for all j when $\mu_k = 0$, and that f is s-times differentiable with derivatives up to s-th order bounded by η. Then, if $\max_{k \geq 1, \mu_k \neq 0} 1/(k^{2s}\mu_k) < C$, we have $f \in \mathcal{H}$ with $\|f\|_{\mathcal{H}}^2 \leq C'\eta^2$, with $C' = 1/\mu_0 + C$ if $\mu_0 \neq 0$, or $C' = C$ if $\mu_0 = 0$.*

Indeed, under the stated conditions, we have

$$\begin{aligned}
\|f\|_{\mathcal{H}}^2 = \sum_{k=0}^{\infty} \sum_{j=1}^{N(p,k)} \frac{a_{k,j}^2}{\mu_k} &\leq \frac{a_{0,1}^2}{\mu_0} + \max_{k \geq 1, \mu_k \neq 0} 1/(k^{2s}\mu_k) \sum_{k,j} a'^2_{k,j} \\
&\leq \frac{1}{\mu_0}\eta^2 + C\|(-\Delta)^{s/2} f\|_{L^2(\mathbb{S})}^2 \leq C'\eta^2.
\end{aligned}$$

Mercer decompositions for arc-cosine kernels. The values of μ_k for the arc-cosine kernels of degree 0 and 1, κ_0 and κ_1, can be immediately obtained from the results of Bach (2017a), and we provide them here. For any non-negative integer $\alpha \geq 0$, Bach (2017a) considers the positively homogeneous activation $\sigma_\alpha(u) = (u)_+^\alpha$ and derives the following quantities for $k \geq 0$:

$$\lambda_{\alpha,k} = \frac{\omega_{p-2}}{\omega_{p-1}} \int_{-1}^{1} \sigma_\alpha(t) P_k(t)(1-t^2)^{(p-3)/2} dt.$$

This can be used to derive the decompositions of the arc-cosine kernels introduced in Section 4.2, which are defined using expectations on Gaussian variables, but can be expressed using expectations on the sphere as follows:

$$\begin{aligned}
\kappa_\alpha(\langle x, y\rangle) &= 2\,\mathbb{E}_{w\sim\mathcal{N}(0,1)}[\sigma_\alpha(\langle w, x\rangle)\sigma_\alpha(\langle w, y\rangle)] \\
&= 2\,\mathbb{E}_{w\sim\mathcal{N}(0,1)}[\|w\|^{2\alpha}\sigma_\alpha(\langle w/\|w\|, x\rangle)\sigma_\alpha(\langle w/\|w\|, y\rangle)] \\
&= 2\,\mathbb{E}_{w\sim\mathcal{N}(0,1)}[\|w\|^{2\alpha}] \int \sigma_\alpha(\langle w, x\rangle)\sigma_\alpha(\langle w, y\rangle) d\tau(w),
\end{aligned}$$

where we used α-homogeneity of σ_α and the rotational symmetry of the normal distribution, which implies that $w/\|w\|$ is uniformly distributed on the sphere, and independent from $\|w\|$.

For a fixed w, we have the decomposition

$$\sigma_\alpha(\langle w, x\rangle) = \sum_{k=0}^{\infty} \lambda_{\alpha,k} N(p,k) P_k(\langle w, y\rangle).$$

Then, using the orthogonality of Legendre polynomials (see (4.20) in Appendix 4.A), we have

$$\kappa_\alpha(\langle x, y\rangle) = 2\,\mathbb{E}_{w\sim\mathcal{N}(0,1)}[\|w\|^{2\alpha}] \sum_{k=0}^{\infty} \lambda_{\alpha,k}^2 N(p,k) P_k(\langle x, y\rangle).$$

For $\alpha = 0, 1$, this yields decompositions (4.10) of κ_0, κ_1 with $\mu_{0,k} = 2\lambda_{0,k}^2$ and $\mu_{1,k} = 2p\lambda_{1,k}^2$. Bach (2017a) then shows the following result on the decomposition of σ_α, which we translate to decompositions of kernel functions κ_α.

Lemma 4.6 (Decomposition of σ_α and κ_α (Bach, 2017a))**.** *For the activation σ_α on the p–1 sphere, we have*

- $\lambda_{\alpha,k} \neq 0$ *if* $k \leq \alpha$*;*
- $\lambda_{\alpha,k} = 0$ *if* $k > \alpha$ *if* $k = \alpha \mod 2$*;*
- $|\lambda_{\alpha,k}| \sim C_\lambda(p,\alpha) k^{-p/2-\alpha}$ *otherwise, for some constant $C_\lambda(p,\alpha)$ depending on p and α.*

For $\alpha \in \{0, 1\}$, the eigenvalues for the corresponding kernel κ_α then satisfy

- $\mu_{\alpha,k} > 0$ *if* $k \leq \alpha$*;*
- $\mu_{\alpha,k} = 0$ *if* $k > \alpha$ *if* $k = \alpha \mod 2$*;*

- $\mu_{\alpha,k} \sim C_\mu(p,\alpha)k^{-p-2\alpha}$ *otherwise, with* $C_\mu(p,\alpha) = 2p^\alpha C_\lambda(p,\alpha)^2$.

Note that the zero eigenvalues imply that a function f of the form (4.12) must have $a_{k,j} = 0$ for $k > \alpha$ and $k = \alpha \mod 2$ in order to be in the RKHS for κ_α (note that adding a bias may prevent such zero eigenvalues, see, *e.g.*, Basri et al., 2019). A sufficient condition for this to hold is that f is even (resp. odd) when α is odd (resp. even); see Bach (2017a).

Mercer decomposition for the NTK. The next proposition gives the Mercer decomposition of the NTK in (4.4), which hereafter will be denoted $\kappa(\langle x, y\rangle)$ for $x, y \in \mathbb{S}^{p-1}$, where κ is given by

$$\kappa(u) = u\kappa_0(u) + \kappa_1(u).$$

Proposition 4.7 (Mercer decomposition of ReLU NTK)**.** *For any* $x, y \in \mathbb{S}^{p-1}$*, we have the following decomposition of the NTK* κ*:*

$$\kappa(\langle x, y\rangle) = \sum_{k=0}^{\infty} \mu_k \sum_{j=1}^{N(p,k)} Y_{k,j}(x)Y_{k,j}(y), \tag{4.14}$$

where $Y_{k,j}, j = 1, \dots, N(p,k)$ *are spherical harmonic polynomials of degree* k*, and the non-negative eigenvalues* μ_k *satisfy* $\mu_0, \mu_1 > 0$*,* $\mu_k = 0$ *if* $k = 2j+1$ *with* $j \geq 1$*, and otherwise* $\mu_k \sim C(p)k^{-p}$ *as* $k \to \infty$*, with* $C(p)$ *a constant depending only on* p*. Then, the RKHS is described by:*

$$\mathcal{H} = \left\{ f = \sum_{k\geq 0, \mu_k \neq 0} \sum_{j=1}^{N(p,k)} a_{k,j} Y_{k,j}(\cdot) \quad s.t. \quad \|f\|_{\mathcal{H}}^2 := \sum_{k\geq 0, \mu_k \neq 0} \sum_{j=1}^{N(p,k)} \frac{a_{k,j}^2}{\mu_k} < \infty \right\}. \tag{4.15}$$

Proof. Using (4.11) and the recurrence relation (4.21) in Appendix 4.A for Legendre polynomials, as well as $tP_0(t) = P_1(t)$, we have

$$\begin{aligned} \mu_0 &= \mu_{0,1} + \mu_{1,0} \\ \mu_k &= \frac{k}{2k+p-2}\mu_{0,k-1} + \frac{k+p-2}{2k+p-2}\mu_{0,k+1} + \mu_{1,k}, \qquad \text{for } k \geq 1. \end{aligned}$$

By Lemma 4.6, we have the desired properties. □

Note that for the arc-cosine 1 kernel κ_1, we have a faster decay $\mu_k = O(k^{-p-2})$, leading to a "smaller" RKHS (see Lemma 4.6 and Bach (2017a)). Moreover, the k^{-p} asymptotic equivalent comes from the term $u\kappa_0(u)$ in the definition (4.4) of κ, which comes from gradients of first layer weights; the second layer gradients yield κ_1, whose contribution to μ_k becomes negligible for large k. We use an identity also used in the recent paper by Ghorbani et al. (2019) which compares similar kernels in a specific high-dimensional limit for generic activations; in contrast to Ghorbani et al. (2019), we focus on ReLUs and study eigenvalue decays in finite dimension. We note that our decomposition uses a uniform distribution on the sphere, which allows a precise study of eigenvalues and approximation properties of the RKHS using spherical harmonics. When the data distribution is also uniform on the sphere, or absolutely continuous

w.r.t. the uniform distribution, our obtained eigenvalues are closely related to those of integral operators for learning problems, which can determine, *e.g.*, non-parametric rates of convergence (*e.g.*, Caponnetto and De Vito, 2007; Fischer and Steinwart, 2017) as well as degrees-of-freedom quantities for kernel approximation (*e.g.*, Bach, 2017b; Rudi and Rosasco, 2017). Such quantities often depend on the eigenvalue decay of the integral operator (see Section 1.3 in Chapter 1), which can be obtained from μ_k after taking multiplicity into account. This is also related to the rate of convergence of gradient descent in the lazy training regime, which depends on the minimum eigenvalue of the empirical kernel matrix in (Chizat et al., 2019; Du et al., 2019a,b).

Regularity conditions and approximation of Lipschitz functions. We now provide sufficient conditions for a function $f : \mathbb{S}^{p-1} \to \mathbb{R}$ to be in $\mathcal{H}$, as well as rates of approximation of Lipschitz functions on the sphere, adapting results of Bach (2017a, specifically Proposition 2 and 3) to our NTK setting.

Corollary 4.8 (Sufficient condition for $f \in \mathcal{H}$)**.** *Let $f : \mathbb{S}^{p-1} \to \mathbb{R}$ be an even function such that all i-th order derivatives exist and are bounded by η for $0 \le i \le s$, with $s \ge p/2$. Then $f \in \mathcal{H}$ with $\|f\|_{\mathcal{H}} \le C(p)\eta$, where $C(p)$ is a constant that only depends on p.*

Corollary 4.9 (Approximation of Lipschitz functions)**.** *Let $f : \mathbb{S}^{p-1} \to \mathbb{R}$ be an even function such that $f(x) \le \eta$ and $|f(x) - f(y)| \le \eta\|x - y\|$, for all $x, y \in \mathbb{S}^{p-1}$. There is a function $g \in \mathcal{H}$ with $\|g\|_{\mathcal{H}} \le \delta$, where δ is larger than a constant depending only on p, such that*

$$\sup_{x \in \mathbb{S}^{p-1}} |f(x) - g(x)| \le C(p)\eta \left(\frac{\delta}{\eta}\right)^{-1/(p/2-1)} \log\left(\frac{\delta}{\eta}\right).$$

Proof sketch. Bach (2017a, Appendix D) defines candidate functions $g : \mathbb{S}^{p-1}$ from functions $p \in L^2(\mathbb{S}^{p-1})$ as $g(x) = Tp(x) := \int p(w)\sigma_\alpha(w^\top x)d\tau(w)$, with RKHS norm (denoted $\gamma_2(g)$ in (Bach, 2017a)) given by the smallest $\|p\|_{L^2}$ for p such that $g = Tp$. In our case with the NTK, we may simply consider the operator $\Sigma^{1/2}$ instead (the self-adjoint square root of the integral operator Σ of κ, using notations from Bach (2017b); see also Cucker and Smale (2002)), which simply multiplies each fourier coefficient $a_{k,j}$ in decomposition (4.12) by $\sqrt{\mu_k}$, and obeys the required properties (in fact, T and $\Sigma^{1/2}$ are two different square roots of Σ (Bach, 2017b)).

The proofs can then be adapted directly, by noticing that $\sqrt{\mu_k}$ has the same decay properties as $\lambda_{\alpha,k}$ with $\alpha = 0$. For Corollary 4.8, the key proof ingredients are also provided in Lemma 4.5 above in our framework. □

For both results, there is an improvement over κ_1, for which Corollary 4.8 requires $s \ge p/2 + 1$ bounded derivatives, and Corollary 4.9 leads to a weaker rate in $(\delta/\eta)^{-1/(p/2)}$ (see Bach, 2017a, Propositions 2 and 3, with $\alpha = 1$). These results show that in the over-parameterized regime of the NTK, training multiple layers leads to better approximation properties compared to only training the last layer, which corresponds to using κ_1 instead of κ. In the different regime of "convex neural networks" (*e.g.*, Bach, 2017a; Savarese et al., 2019) where neurons can be selected with a sparsity-promoting penalty, the approximation rates shown by Bach (2017a) for ReLU networks are also weaker than for the NTK in the worst case (though the regime presents benefits in terms of adaptivity), suggesting that perhaps in some situations the "lazy" regime of the NTK could be preferred over the regime where neurons are selected using sparsity.

Homogeneous case. When inputs do not lie on the sphere $\mathbb{S}^{p-1}$ but in $\mathbb{R}^p$, the NTK for two-layer ReLU networks takes the form of a homogeneous dot-product kernel (4.3), which defines a different RKHS $\bar{\mathcal{H}}$ that we characterize below in terms of the RKHS $\mathcal{H}$ of the NTK on the sphere.

Proposition 4.10 (RKHS of the homogeneous NTK)**.** *The RKHS $\bar{\mathcal{H}}$ of the kernel $K(x,x') = \|x\|\|x'\|\kappa(\langle x,x'\rangle/\|x\|\|x'\|)$ on $\mathbb{R}^p$ consists of functions of the form $f(x) = \|x\|g(x/\|x\|)$ with $g \in \mathcal{H}$, where $\mathcal{H}$ is the RKHS on the sphere, and we have $\|f\|_{\bar{\mathcal{H}}} = \|g\|_{\mathcal{H}}$.*

Proof of Proposition 4.10. The kernel K can be written as

$$K(x,x') = \langle \|x\|\Phi\left(\frac{x}{\|x\|}\right), \|x'\|\Phi\left(\frac{x'}{\|x'\|}\right)\rangle_{\mathcal{H}},$$

where $\Phi(\cdot)$ is the kernel mapping of the kernel κ on the sphere. Then the RKHS $\bar{\mathcal{H}}$ can be characterized by the following classical result—see, *e.g.*, Theorem 1.3 in Chapter 1:

$$\bar{\mathcal{H}} = \{\underbrace{x \mapsto \langle g, \|x\|\Phi\left(\frac{x}{\|x\|}\right)\rangle_{\mathcal{H}}}_{=:f_g} : g \in \mathcal{H}\}$$
$$\|f_g\|_{\bar{\mathcal{H}}} = \inf\{\|g'\|_{\mathcal{H}} : g' \in \mathcal{H} \text{ s.t. } f_g = f_{g'}\}.$$

Note that the condition $f_g = f_{g'}$ implies in particular that f_g and $f_{g'}$ are equal on the sphere, and thus that $g = g'$, so that the infimum is simply equal to $\|g\|_{\mathcal{H}}$. This concludes the proof. □

Note that while such a restriction to homogeneous functions may be limiting, one may easily obtain non-homogeneous functions by considering an augmented variable $z = (x^\top, R)^\top$ and defining $f(x) = \|z\|g(z/\|z\|)$, where g is now defined on the p-sphere $\mathbb{S}^p$. When inputs are in a ball of radius R, this reformulation preserves regularity properties (see Bach, 2017a, Section 3).

4.3.3 Smoothness with other activations

In this section, we look at smoothness of two-layer networks with different activation functions. Following the derivation for the ReLU in Section 4.2.1, the NTK for a general activation σ is given by

$$K_\sigma(x,x') = \langle x,x'\rangle\, \mathbb{E}_{w\sim\mathcal{N}(0,1)}[\sigma'(\langle w,x\rangle)\sigma'(\langle w,x'\rangle)] + \mathbb{E}_{w\sim\mathcal{N}(0,1)}[\sigma(\langle w,x\rangle)\sigma(\langle w,x'\rangle)].$$

We then have the following the following result.

Proposition 4.11 (Lipschitzness for smooth activations)**.** *Assume that σ is twice differentiable and that the quantities $\gamma_j := \mathbb{E}_{u\sim\mathcal{N}(0,1)}[(\sigma^{(j)}(u))^2]$ for $j = 0,1,2$ are bounded, with $\gamma_0 > 0$. Then, for x, y on the unit sphere, the kernel mapping Φ_σ of K_σ satisfies*

$$\|\Phi_\sigma(x) - \Phi_\sigma(y)\| \leq \sqrt{(\gamma_0+\gamma_1)\max\left(1, \frac{2\gamma_1+\gamma_2}{\gamma_0+\gamma_1}\right)} \cdot \|x-y\|.$$

The proof uses results by Daniely et al. (2016) on relationships between activations and the corresponding kernels, as well as smoothness results for dot-product kernels in Chapter 2.3 (the proof is given in Appendix 4.C.3). If, for instance, we consider the exponential activation $\sigma(u) = e^{u-2}$, we have $\gamma_j = 1$ for all j (using results of Daniely et al. (2016)), so that the kernel mapping is Lipschitz with constant $\sqrt{3}$. For the soft-plus activation $\sigma(u) = \log(1 + e^u)$, we may evaluate the integrals numerically, obtaining $(\gamma_0, \gamma_1, \gamma_2) \approx (2.31, 0.74, 0.11)$, so that the kernel mapping is Lipschitz with constant ≈ 1.75.

4.4 Deep Convolutional Networks

In this section, we study smoothness and stability properties of the NTK kernel mapping for convolutional networks with ReLU activations. In order to properly define deformations, we consider continuous signals $x(u)$ in $L^2(\mathbb{R}^d)$ instead of $\ell^2(\mathbb{Z}^d)$ (*i.e.*, we have $\|x\|^2 := \int \|x(u)\|^2 du < \infty$), following Chapter 2 and Mallat (2012). The goal of deformation stability guarantees is to ensure that the data representation (in this case, the kernel mapping Φ) does not change too much when the input signal is slightly deformed, for instance with a small translation or rotation of an image—a useful inductive bias for natural signals. For a C^1-diffeomorphism $\tau : \mathbb{R}^d \to \mathbb{R}^d$, denoting $L_\tau x(u) = x(u - \tau(u))$ the action operator of the diffeomorphism, we will show a guarantee of the form

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq (\omega(\|\nabla\tau\|_\infty) + C\|\tau\|_\infty)\|x\|,$$

where $\|\nabla\tau\|_\infty$ is the maximum operator norm of the Jacobian $\nabla\tau(u)$ over $\mathbb{R}^d$, $\|\tau\|_\infty = \sup_u |\tau(u)|$, ω is an increasing function and C a positive constant. The second term controls translation invariance, and C typically decreases with the scale of the last pooling layer (σ_n below), while the first term controls deformation stability, since $\|\nabla\tau\|_\infty$ measures the "size" of deformations. The function $\omega(t)$ is typically a linear function of t in other settings, as discussed in Section 1.5.2 of Chapter 1, but here we will obtain a faster growth of order $\sqrt{t}$ for small t, due to the weaker smoothness that arises from the arc-cosine 0 kernel mappings.

Properties of the operators. In this continuous setup, P^k is now given for a signal $x \in L^2$ by $P^k x(u) = \lambda(S_k)^{-1/2}(x(u+v))_{v \in S_k}$, where λ is the Lebesgue measure. We then have $\|P^k x\| = \|x\|$, and considering normalized Gaussian pooling filters, we have $\|A^k x\| \leq \|x\|$ by Young's inequality, as in Chapter 2. The non-linear operator M is defined point-wise analogously to (4.8), and satisfies $\|M(x,y)\|^2 = \|x\|^2 + \|y\|^2$. We thus have that the feature maps in the continuous analog of the NTK construction in Proposition 4.2 are in L^2 as long as x is in L^2. Note that this does not hold for some smooth activations, where $\|M(x,y)(u)\|$ may be a positive constant even when $x(u) = y(u) = 0$, leading to unbounded L^2 norm for $M(x,y)$. The next lemma studies the smoothness of M, extending results from Section 4.3.1 to signals in L^2.

Lemma 4.12 (Smoothness of operator M)**.** *For two signals $x, y \in L^2(\mathbb{R}^d)$, we have*

$$\|M(x,y) - M(x',y')\| \leq \sqrt{\min(\|y\|, \|y'\|)\|x - x'\|} + \|x - x'\| + \|y - y'\|. \tag{4.16}$$

Assumptions on architecture. Following Chapter 2, we introduce an initial pooling layer A^0, corresponding to an anti-aliasing filter, which is necessary to allow stability and is a reasonable assumption given that in practice the inputs are discrete signals, for which high frequencies have typically been filtered by an acquisition device. Thus, we consider the kernel representation $\Phi_n(x) := \Phi(A_0 x)$, with Φ as in Proposition 4.2. We also assume that patch sizes are controlled by the scale of pooling filters, that is

$$\sup_{v \in S_k} |v| \leq \beta \sigma_{k-1}, \tag{4.17}$$

for some constant β, where σ_{k-1} is the scale of the pooling operation A^{k-1}, which typically increases exponentially with depth, corresponding to a fixed downsampling factor at each layer in the discrete case. By a simple induction, we can show the following.

Lemma 4.13 (Norm and smoothness of Φ_n)**.** *We have* $\|\Phi_n(x)\| \leq \sqrt{n+1}\|x\|$*, and*

$$\|\Phi_n(x) - \Phi_n(x')\| \leq (n+1)\|x - x'\| + O(n^{5/4})\sqrt{\|x\|\|x - x'\|}.$$

Deformation stability bound. We now present our main guarantee on deformation stability for the NTK kernel mapping.

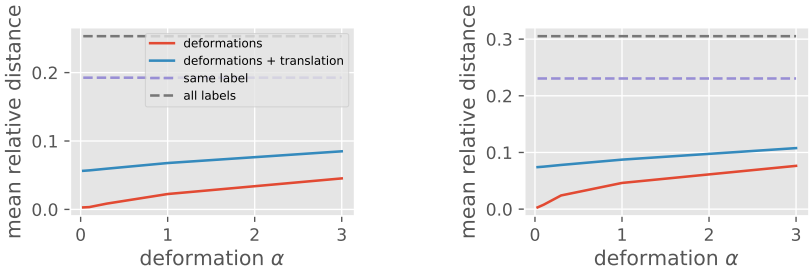
Proposition 4.14 (Stability of NTK)**.** *Let* $\Phi_n(x) = \Phi(A_0 x)$*, and assume* $\|\nabla \tau\|_\infty \leq 1/2$*. We have the following stability bound:*

$$\begin{aligned}&\|\Phi_n(L_\tau x) - \Phi_n(x)\| \\ &\qquad \leq \left(C(\beta)^{1/2} C n^{7/4} \|\nabla \tau\|_\infty^{1/2} + C(\beta) C' n^2 \|\nabla \tau\|_\infty + \sqrt{n+1}\frac{C''}{\sigma_n}\|\tau\|_\infty \right) \|x\|,\end{aligned}$$

where C, C', C'' *are constants depending only on* d*, and* $C(\beta)$ *also depends on* β *defined in* (4.17)*.*

The proof is given in Appendix 4.C. Compared to the bound in Chapter 2, the first term shows weaker stability due to faster growth with $\|\nabla \tau\|_\infty$, which comes from (4.16). The dependence in n is also poorer (n^2 instead of n), however note that in contrast to Chapter 2, the norm and smoothness constants of $\Phi_n(x)$ in Lemma 4.13 grow with n here, partially explaining this gap. We also note that as in Chapter 2.3, choosing small β (*i.e.*, small patches in a discrete setting) is more helpful to improve stability than a small number of layers n, given that $C(\beta)$ increases with β as β^{d+1}, while n typically decreases with β as $1/\log(\beta)$ when one seeks a fixed target level of translation invariance (see Section 2.3.2 in Chapter 2).

By fixing weights of all layers but the last, we would instead obtain feature maps of the form $A^n x_n$ (using notation from Proposition 4.2), which satisfy the improved stability guarantee of Chapter 2. The question of approximation for the deep convolutional case is more involved and left for future work, but it is reasonable to expect that the RKHS for the NTK is at least as large as that of the simpler kernel with fixed layers before the last, given that the latter appears as one of the terms in the NTK. This again hints at a tradeoff between stability and approximation, suggesting that one may be able to learn less stable but more discriminative functions in the NTK regime by training all layers.



(a) CKN with arc-cosine 1 kernels (b) NTK

Figure 4.1: Average relative representation distance for CKN and NTK convolutional kernels, on the same MNIST digit data considered in Section 2.3.4, similar to Figure 2.3; see Chapter 2 for details on the experimental setup. The figures consider deformations of varying size $\alpha \in \{0.01, 0.03, 0.1, 0.3, 1, 3\}$.

Numerical experiments. We now study numerically the stability of (exact) kernel mapping representations for convolutional networks with 2 hidden convolutional layers on the same images of digits as considered in Section 2.3.4 of Chapter 2, that is, MNIST digits along with their deformations from the infinite MNIST dataset of Loosli et al. (2007). We consider both the convolutional kernel of Chapter 2 with arc-cosine kernels of degree 1 on patches (corresponding to the kernel obtained when only training the last layer and keeping previous layers fixed), denoted CKN, to the NTK. The kernel evaluations are computed in C++ using dynamic programming; our implementation is available at `https://github.com/albietz/ckn_kernel`.

Figure 4.1 shows the resulting average distances, when considering collections of digits and deformations thereof. In particular, we find that for small deformations (small α), the distance to the original image tends to grow more quickly for the NTK compared to the CKN, as the theory suggests (a square-root growth rate rather than a linear one). Note also that the relative distances are generally larger for the NTK than for the CKN, suggesting the CKN may be smoother.

4.5 Discussion

In this chapter, we have studied the inductive bias of the "lazy training" regime for over-parameterized neural networks, by considering the neural tangent kernel of different architectures, and analyzing properties of the corresponding RKHS, which characterizes the functions that can be learned efficiently in this regime. We find that the NTK for ReLU networks has better approximation properties compared to other neural network kernels, but weaker smoothness properties, although these can still guarantee a form of stability to deformations for CNN architectures, providing an important inductive bias for natural signals. While these properties may help obtain better performance when large amounts of data are available, they can also lead to a poorer estimation error when data is scarce, a setting in which smoother kernels or better regularization strategies may be helpful.

It should be noted that while our study of functions in the RKHS may determine what target functions can be learned by over-parameterized networks, the obtained networks

with finite neurons do not belong to the same RKHS, and hence may be less stable than such target functions, at least outside of the training data, due to approximations both in the linearization (4.1) and between the finite neuron and limiting kernels. Finally, we note that while this "lazy" regime is interesting and could partly explain the success of deep learning methods, it does not explain, for instance, the common behavior in early layers where neurons move to select useful features in the data, such as Gabor filters, as pointed out by Chizat et al. (2019). In particular, such a behavior might provide better statistical efficiency by adapting to simple structures in the data (see, *e.g.*, Bach, 2017a), something which is not captured in a kernel regime like the NTK. It would be interesting to study inductive biases in a regime somewhere in between, where neurons may move at least in the first few layers.

Appendix

4.A Background on spherical harmonics

In this section, we provide some background on spherical harmonic analysis needed for our study of the RKHS and its approximation properties. See Efthimiou and Frye (2014); Atkinson and Han (2012) for references, as well as Bach (2017a, Appendix D). We consider inputs on the $p-1$ sphere $\mathbb{S}^{p-1} = \{x \in \mathbb{R}^p, \|x\| = 1\}$.

We denote by $Y_{k,j}(x)$, $j = 1, \dots, N(p,k)$, the spherical harmonics of degree k on $\mathbb{S}^{p-1}$, where $N(p,k) = \frac{2k+p-2}{k}\binom{k+p-3}{p-2}$. They form an orthonormal basis of $L^2(\mathbb{S}^{p-1}, d\tau)$, where τ is the uniform measure on the sphere. The index k plays the role of an integer frequency, as in Fourier. We have the addition formula

$$\sum_{j=1}^{N(p,k)} Y_{k,j}(x) Y_{k,j}(y) = N(p,k) P_k(\langle x, y \rangle), \tag{4.18}$$

where P_k is the k-th Legendre polynomial in dimension p (also known as Gegenbauer polynomials), given by the Rodrigues formula:

$$P_k(t) = (-1/2)^k \frac{\Gamma(\frac{p-1}{2})}{\Gamma(k + \frac{p-1}{2})} (1-t^2)^{(3-p)/2} \left(\frac{d}{dt}\right)^k (1-t^2)^{k+(d-3)/2}.$$

The polynomials P_k are orthogonal in $L^2([-1,1], d\nu)$ where the measure $d\nu$ is given by $d\nu(t) = (1-t^2)^{(p-3)/2} dt$, and we have

$$\int_{-1}^{1} P_k^2(t)(1-t^2)^{(p-3)/2} dt = \frac{\omega_{p-1}}{\omega_{p-2}} \frac{1}{N(p,k)}, \tag{4.19}$$

where $\omega_{d-1} = \frac{2\pi^{d/2}}{\Gamma(d/2)}$ denotes the surface of the sphere $\mathbb{S}^{d-1}$ in d dimensions. Using the addition formula (4.18) and orthogonality of spherical harmonics, we can show

$$\int P_j(\langle w, x \rangle) P_k(\langle w, y \rangle) d\tau(w) = \frac{\delta_{jk}}{N(p,k)} P_k(\langle x, y \rangle) \tag{4.20}$$

Further, we have the recurrence relation (Efthimiou and Frye, 2014, Eq. 4.36)

$$t P_k(t) = \frac{k}{2k+p-2} P_{k-1}(t) + \frac{k+p-2}{2k+p-2} P_{k+1}(t), \tag{4.21}$$

for $k \geq 1$, and for $k = 0$ we simply have $tP_0(t) = P_1(t)$.

The Funk-Hecke formula is helpful for computing Fourier coefficients in the basis of spherical harmonics in terms of Legendre polynomials: for any $j = 1, \dots, N(p,k)$, we have

$$\int f(\langle x, y\rangle) Y_{k,j}(y) d\tau(y) = \frac{\omega_{p-2}}{\omega_{p-1}} Y_{k,j}(x) \int_{-1}^{1} f(t) P_k(t) (1-t^2)^{(p-3)/2} dt. \tag{4.22}$$

4.B Proofs of NTK derivations

4.B.1 Proof of Lemma 4.1

Proof of Lemma 4.1. By induction, using (4.6) and (4.7) and the corresponding definitions of φ_1, φ_0, we can write

$$\begin{aligned} 2\,\mathbb{E}_{(u,v)\sim\mathcal{N}(0,B_k)}[\sigma(u)\sigma(v)] &= \langle \varphi_1(\Psi_{k-1}(x)), \varphi_1(\Psi_{k-1}(x'))\rangle \\ 2\,\mathbb{E}_{(u,v)\sim\mathcal{N}(0,B_k)}[\sigma'(u)\sigma'(v)] &= \langle \varphi_0(\Psi_{k-1}(x)), \varphi_0(\Psi_{k-1}(x'))\rangle. \end{aligned}$$

The result follows by using the following relation, given three pairs of vectors (x, x'), (y, y') and (z, z') in arbitrary Hilbert spaces:

$$\langle x, x'\rangle + \langle y, y'\rangle\langle z, z'\rangle = \langle \begin{pmatrix} y \otimes z \\ x \end{pmatrix}, \begin{pmatrix} y' \otimes z' \\ x' \end{pmatrix} \rangle$$

□

4.B.2 Proof of Proposition 4.2 (NTK for CNNs)

In this section, we will denote by x_k, y_k (resp x'_k, y'_k) the feature maps associated to an input x (resp x'), as defined in Proposition 4.2. We follow the proofs of Jacot et al. (Jacot et al., 2018, Proposition 1 and Theorem 1).

We begin by proving the following lemma, which characterizes the Gaussian process behavior of the pre-activations $\tilde{a}_i^k[u]$, seen as a function of x and u, in the over-parameterization limit.

Lemma 4.15. *As $m_1, \dots, m_{n-1} \to \infty$, the pre-activations $\tilde{a}_i^k[u]$ for $k = 1, \dots, n$ tend (in law) to i.i.d. centered Gaussian processes with covariance*

$$\Sigma^k(x, u; x', u') = \langle x_k[u], x'_k[u']\rangle. \tag{4.23}$$

Proof. We show this by induction. For $k = 1$, $\tilde{a}_i^1[u]$ is clearly Gaussian, and we have

$$\begin{aligned} \Sigma^1(x, u; x', u') &= \mathbb{E}[\tilde{a}_i^1[u]\tilde{a}_i'^1[u']] \\ &= \mathbb{E}[(W^1 P^1 x[u])_i (W^1 P^1 x'[u'])_i]. \end{aligned}$$

Writing $W_{ij}^k \in \mathbb{R}^{|S_k|}$ the vector of weights for the filter associated to the input feature map j and output feature map i, we have $(W^1 P^1 x[u])_i = \sum_{j=1}^{m_1} W_{ij}^{1\top} P^1 x_j[u]$. Then we

have

$$\begin{aligned}\Sigma^k(x,u;x',u') &= \sum_{j,j'} \mathbb{E}[W_{ij}^{1\top} P^1 x_j[u] P^1 x'_{j'}[u']^\top W_{ij'}^1] \\ &= \sum_{j,j'} \mathrm{Tr}(\mathbb{E}[W_{ij'}^1 W_{ij}^{1\top}] P^1 x_j[u] P^1 x'_{j'}[u']^\top) \\ &= \sum_{j} \mathrm{Tr}(P^1 x_j[u] P^1 x'_j[u']^\top) = \langle P^1 x[u], P^1 x'[u']\rangle = \langle P^1 x_0[u], P^1 x'_0[u']\rangle,\end{aligned}$$

by noticing that $\mathbb{E}[W_{ij'}^1 W_{ij}^{1\top}] = \delta_{j,j'} I_{|S_1|}$.

Now, for $k \geq 2$, we have by similar arguments that conditioned on a^{k-1}, $\tilde{a}_i^k[u]$ is Gaussian, with covariance

$$\mathbb{E}[\tilde{a}_i^k[u]\tilde{a}_i'^k[u']|a^{k-1}, a'^{k-1}] = \frac{2}{m_{k-1}} \sum_j \langle P^k a_j^{k-1}[u], P^k a_j'^{k-1}[u']\rangle.$$

By the inductive hypothesis, $\tilde{a}_j^{k-1}[u]$ as a function of x and u tend to Gaussian processes in the limit $m_1, \ldots, m_{k-2} \to \infty$. By the law of large numbers, we have, as $m_{k-1} \to \infty$,

$$\begin{aligned}&\mathbb{E}[\tilde{a}_i^k[u]\tilde{a}_i'^k[u']|a^{k-1}, a'^{k-1}] \\ &\qquad \to \Sigma^k(x,u;x',u') := 2\,\mathbb{E}_{f\sim GP(0,\Sigma^{k-1})}[\langle P^k A^{k-1}\sigma(f(x))[u], P^k A^{k-1}\sigma(f(x'))[u']\rangle].\end{aligned}$$

Since this covariance is deterministic, the pre-activations $\tilde{a}_i^k[u]$ are also unconditionally a Gaussian process in the limit, with covariance Σ^k.

Now it remains to show that

$$\begin{aligned}&2\,\mathbb{E}_{f\sim GP(0,\Sigma^{k-1})}[\langle P^k A^{k-1}\sigma(f(x))[u], P^k A^{k-1}\sigma(f(x'))[u']\rangle] \\ &\qquad = \langle P^k A^{k-1}\varphi_1(x_{k-1})[u], P^k A^{k-1}\varphi_1(x'_{k-1})[u']\rangle.\end{aligned}$$

Notice that by linearity of P^k and A^{k-1}, it suffices to show

$$\begin{aligned}2\,\mathbb{E}_{f\sim GP(0,\Sigma^{k-1})}[\sigma(f(x))[v]\sigma(f(x'))[v']] &= \langle \varphi_1(x_{k-1})[v], \varphi_1(x'_{k-1})[v']\rangle \\ &= \|x_{k-1}[v]\|\|x'_{k-1}[v']\|\kappa_1\left(\frac{\langle x_{k-1}[v], x_{k-1}[v']\rangle}{\|x_{k-1}[v]\|\|x'_{k-1}[v']\|}\right),\end{aligned}$$

for any v, v' (the last equality follows from the definition of φ_1). Noting that the tuple $(\sigma(f(x))[v], \sigma(f(x'))[v']) = (\sigma(f(x)[v]), \sigma(f(x')[v']))$ when $f \sim GP(0, \Sigma^{k-1})$ has Gaussian distribution with zero mean and covariance $\begin{pmatrix}\Sigma^{k-1}(x,v;x,v) & \Sigma^{k-1}(x,v;x',v') \\ \Sigma^{k-1}(x',v';x',v') & \Sigma^{k-1}(x',v';x',v')\end{pmatrix}$, the results follow from (4.6) and (4.23) for k–1 by the inductive hypothesis. □

We now state and prove a lemma which covers the recursion in the NTK for convolutional layers (*i.e.*, up to the last fully-connected layer).

Lemma 4.16. *As $m_1, \ldots, m_{n-1} \to \infty$, the gradients of the pre-activations, $\nabla_\theta \tilde{a}_i^k[u]$, for $k = 1, \ldots, n$ satisfy*

$$\langle \nabla_\theta \tilde{a}_i^k[u], \nabla_\theta \tilde{a}_{i'}'^k[u']\rangle \to \delta_{i,i'}\tilde{\Gamma}_\infty^k(x,u;x',u') = \delta_{i,i'}\langle y_k[u], y'_k[u']\rangle.$$

Proof. We prove this by induction. For $k = 1$, denoting by W_i^1 the ith row of W^1, we have

$$\begin{aligned}\langle \nabla_\theta \tilde{a}_i^1[u], \nabla_\theta \tilde{a}_{i'}'^1[u'] \rangle &= \langle \nabla_{W^1} (W^1 P^1 x[u])_i, \nabla_{W^1} (W^1 P^1 x'[u'])_{i'} \rangle \\ &= \sum_s \langle \nabla_{W_s^1} W_i^1 P^1 x[u], \nabla_{W_s^1} W_{i'}^1 P^1 x'[u'] \rangle \\ &= \delta_{i,i'} \langle P^1 x[u], P^1 x'[u'] \rangle.\end{aligned}$$

For $k \geq 2$, assume the result holds up to k–1. We have

$$\langle \nabla_\theta \tilde{a}_i^k[u], \nabla_\theta \tilde{a}_{i'}'^k[u'] \rangle = \langle \nabla_{W^k} \tilde{a}_i^k[u], \nabla_{W^k} \tilde{a}_{i'}'^k[u'] \rangle + \langle \nabla_{W^{1:k-1}} \tilde{a}_i^k[u], \nabla_{W^{1:k-1}} \tilde{a}_{i'}'^k[u'] \rangle.$$

For the first term, we have, as in the $k = 1$ case,

$$\begin{aligned}\langle \nabla_{W^k} \tilde{a}_i^k[u], \nabla_{W^k} \tilde{a}_{i'}'^k[u'] \rangle &= \frac{2\delta_{i,i'}}{m_{k-1}} \langle P^k a^{k-1}[u], P^k a'^{k-1}[u'] \rangle \\ &= \frac{2\delta_{i,i'}}{m_{k-1}} \sum_j \langle P^k a_j^{k-1}[u], P^k a_j'^{k-1}[u'] \rangle \\ &= \frac{2\delta_{i,i'}}{m_{k-1}} \sum_j \langle P^k A^{k-1} \sigma(\tilde{a}_j^{k-1})[u], P^k A^{k-1} \sigma(\tilde{a}_j'^{k-1})[u'] \rangle.\end{aligned}$$

When $m_1, \ldots, m_{k-2} \to \infty$, $\tilde{a}_j^{k-1}[u]$ tends to a Gaussian process with covariance Σ^{k-1} by Lemma 4.15, and when $m_{k-1} \to \infty$, the quantity above converges to its expectation:

$$\begin{aligned}\langle \nabla_{W^k} \tilde{a}_i^k[u], \nabla_{W^k} \tilde{a}_{i'}'^k[u'] \rangle &\to 2\delta_{i,i'} \, \mathbb{E}_{f \sim GP(0, \Sigma^{k-1})} [\langle P^k A^{k-1} \sigma(f(x))[u], P^k A^{k-1} \sigma(f(x))[u'] \rangle] \\ &= \delta_{i,i'} \langle P^k A^{k-1} \varphi_1(x_{k-1})[u], P^k A^{k-1} \varphi_1(x'_{k-1})[u'] \rangle,\end{aligned}$$

by using similar arguments to the proof of Lemma 4.15.

For the second term, identifying all parameters $W^{1:k-1}$ with a vector $\hat{\theta} \in \mathbb{R}^q$, we have by linearity and the chain rule:

$$\begin{aligned}\sqrt{m_{k-1}} \nabla_{\hat{\theta}} \tilde{a}_i^k[u] &= \sum_j \nabla_{\hat{\theta}} (W_{ij}^{k\top} P^k A^{k-1} \tilde{a}_j^{k-1}[u]) \\ &= \sum_j P^k A^{k-1} y_j^{k-1}[u]^\top \cdot W_{ij}^k \in \mathbb{R}^q,\end{aligned}$$

where $y_j^{k-1}[u] := \sqrt{2}\sigma'(\tilde{a}_j^{k-1}[u]) \nabla_{\hat{\theta}} \tilde{a}_j^{k-1}[u] \in \mathbb{R}^q$. Here we have identified $P^k A^{k-1} y_j^{k-1}[u]^\top$ with a matrix in $\mathbb{R}^{q \times |S_k|}$, where columns are given by $|S_k|^{-1/2} A^{k-1} y_j^{k-1}[u+v] \in \mathbb{R}^q$, indexed by $v \in S_k$. We thus have

$$\langle \nabla_{W^{1:k-1}} \tilde{a}_i^k[u], \nabla_{W^{1:k-1}} \tilde{a}_{i'}'^k[u'] \rangle = \frac{1}{m_{k-1}} \sum_{j,j'} W_{i,j}^{k\top} \underbrace{(P^k A^{k-1} y_j^{k-1}[u] \cdot P^k A^{k-1} y_{j'}'^{k-1}[u']^\top)}_{=: \Pi^{j,j'} \in \mathbb{R}^{|S_k| \times |S_k|}} W_{i',j'}^k.$$

For $v, v' \in S_k$, when $m_1, \ldots, m_{k-2} \to \infty$ and using the inductive hypothesis, we have

$$\begin{aligned}\Pi_{v,v'}^{j,j'} &= \frac{1}{|S_k|} A^{k-1} y_j^{k-1}[u+v]^\top A^{k-1} y_{j'}'^{k-1}[u'+v'] \\ &\to \delta_{j,j'} \bar{\Pi}_{v,v'}^j := \frac{\delta_{j,j'}}{|S_k|} \langle A^{k-1} \gamma_j^{k-1}[u+v], A^{k-1} \gamma_j'^{k-1}[u'+v'] \rangle,\end{aligned}$$

where $\gamma_j^{k-1}[u] := \sqrt{2}\sigma'(\tilde{a}_j^{k-1}[u])y_{k-1}[u]$. Indeed, by linearity it suffices to check that $y_j^{k-1}[u]^\top y_{j'}^{k-1}[u'] = 2\delta_{j,j'}\sigma'(\tilde{a}_j^{k-1}[u])\sigma'(\tilde{a}_{j'}'^{k-1}[u'])\langle y_{k-1}[u], y'_{k-1}[u']\rangle$ for any j, j', u, u', which is true by the inductive hypothesis. In this same limit (with m_{k-1} fixed), we then have

$$\langle \nabla_{W^{1:k-1}} \tilde{a}_i^k[u], \nabla_{W^{1:k-1}} \tilde{a}_{i'}'^k[u']\rangle \to \frac{1}{m_{k-1}} \sum_j W_{i,j}^{k\top} \bar{\Pi}^j W_{i',j}^k.$$

When $m_{k-1} \to \infty$, by the law of large numbers, this quantity converges to its expectation:

$$\langle \nabla_{W^{1:k-1}} \tilde{a}_i^k[u], \nabla_{W^{1:k-1}} \tilde{a}_{i'}'^k[u']\rangle \to \mathrm{Tr}(\mathbb{E}[W_{i',1}^k W_{i,1}^{k\top}]\Pi^\infty) = \delta_{i,i'}\,\mathrm{Tr}(\Pi^\infty),$$

where Π^∞ is given by

$$\Pi^\infty_{v,v'} = \frac{1}{|S_k|}\langle A^{k-1}\gamma_\infty^{k-1}[u+v], A^{k-1}\gamma_\infty'^{k-1}[u'+v']\rangle,$$

with $\gamma_\infty^{k-1}[u] = \varphi_0(x_{k-1}[u]) \otimes y_{k-1}[u]$. Indeed, using Lemma 4.15 and linearity of A^{k-1}, it is enough to check that

$$2\,\mathbb{E}_{f\sim GP(0,\Sigma^{k-1})}[\sigma'(f(x)[u])\sigma'(f(x')[u'])\langle y_{k-1}[u], y'_{k-1}[u']\rangle] = \langle \gamma_\infty^{k-1}[u], \gamma_\infty'^{k-1}[u']\rangle,$$

which holds by definition of φ_0 and Σ^{k-1}.

Finally, notice that

$$\begin{aligned}\mathrm{Tr}(\Pi^\infty) &= \frac{1}{|S_k|}\sum_{v\in S_k}\langle A^{k-1}\gamma_\infty^{k-1}[u+v], A^{k-1}\gamma_\infty'^{k-1}[u'+v]\rangle \\ &= \langle P^k A^{k-1}\gamma_\infty^{k-1}[u], P^k A^{k-1}\gamma_\infty'^{k-1}[u']\rangle.\end{aligned}$$

Thus we have

$$\begin{aligned}\tilde{\Gamma}_\infty(x,u,x',u') &= \langle P^k A^{k-1}\varphi_1(x_{k-1})[u], P^k A^{k-1}\varphi_1(x'_{k-1})[u']\rangle \\ &\quad + \langle P^k A^{k-1}\gamma_\infty^{k-1}[u], P^k A^{k-1}\gamma_\infty'^{k-1}[u']\rangle \\ &= \langle P^k A^{k-1} M(x_{k-1}, y_{k-1})[u], P^k A^{k-1} M(x'_{k-1}, y'_{k-1})[u']\rangle \\ &= \langle y_k[u], y'_k[u']\rangle,\end{aligned}$$

which concludes the proof.

□

Armed with the two above lemmas, we can now prove Proposition 4.2 by studying the gradient of the prediction layer.

Proof of Proposition 4.2. We have

$$\langle \nabla_\theta f(x;\theta), \nabla_\theta f(x';\theta)\rangle = \langle \nabla_{w^{n+1}} f(x;\theta), \nabla_{w^{n+1}} f(x';\theta)\rangle + \langle \nabla_{W^{1:n}} f(x;\theta), \nabla_{W^{1:n}} f(x';\theta)\rangle$$

The first term writes

$$\begin{aligned}\langle \nabla_{w^{n+1}} f(x;\theta), \nabla_{w^{n+1}} f(x';\theta)\rangle &= \frac{2}{m_n}\sum_j \langle a_j^n, a_j'^n\rangle \\ &= \frac{2}{m_n}\sum_j\sum_u \langle A^n\sigma(\tilde{a}_j^n)[u], A^n\sigma(\tilde{a}_j'^n)[u]\rangle\end{aligned}$$

Using similar arguments as in the above proofs and using Lemma 4.15, as $m_1, \dots, m_n \to \infty$, we have

$$\langle \nabla_{w^{n+1}} f(x;\theta), \nabla_{w^{n+1}} f(x';\theta)\rangle \to \sum_u \langle A^n \varphi_1(x_n)[u], A^n \varphi_1(x'_n)[u]\rangle = \langle A^n\varphi_1(x_n), A^n\varphi_1(x'_n)\rangle.$$

For the second term, we have

$$\langle \nabla_{W^{1:n}} f(x;\theta), \nabla_{W^{1:n}} f(x';\theta) = \frac{2}{m_n}\sum_{u,u'}\sum_{j,j'} w_j^{n+1}[u] w_{j'}^{n+1}[u'] \langle \nabla_{W^{1:n}} a_j^n[u], \nabla_{W^{1:n}} a_{j'}'^n[u']\rangle.$$

We can use similar arguments to the proof of Lemma 4.16 to show that when $m_1, \dots, m_n \to \infty$, we have

$$\langle \nabla_{W^{1:n}} f(x;\theta), \nabla_{W^{1:n}} f(x';\theta) \to \sum_{u,u'} \mathbb{E}[w_1^{n+1}[u] w_1^{n+1}[u']] \langle A^n\gamma_n[u], A^n\gamma'_n[u']\rangle = \langle A^n\gamma_n, A^n\gamma'_n\rangle,$$

where $\gamma_n[u] := \varphi_0(x_n[u]) \otimes y_n[u]$.

The final result follows by combining both terms. □

4.C Proofs for Smoothness and Stability to Deformations

4.C.1 Proof of Proposition 4.3

Proof. Using notations from Section 4.2.1, we can write

$$\kappa(u) = u\kappa_0(u) + \kappa_1(u) = \frac{u}{\pi}(\pi - \arccos(u)) + \frac{1}{\pi}(u(\pi - \arccos(u)) + \sqrt{1-u^2}).$$

For $\|x\| = \|y\| = 1$, and denoting $u = \langle x, y\rangle$, we have

$$\begin{aligned}\frac{\|\Phi(x) - \Phi(y)\|^2}{\|x-y\|^2} &= \frac{2\kappa(1) - 2\kappa(u)}{2-2u} \\ &= \frac{\kappa_0(1) - u\kappa_0(u)}{1-u} + \frac{\kappa_1(1) - \kappa_1(u)}{1-u} \\ &\sim_{u\to 1^-} u\kappa_0'(u) + \kappa_0(u) + \kappa_1'(u) \xrightarrow{u\to 1^-} +\infty,\end{aligned}$$

where the equivalent follows from l'Hôpital's rule, and we have $\kappa_0'(u) = 1/\pi\sqrt{1-u^2} \to +\infty$, while $\kappa_0(1) = \kappa_1(1) = \kappa_1'(1) = 1$. It follows that the supremum over x, y is unbounded.

For the second part, fix an arbitrary $L > 0$. We can find x, y such that $\|\Phi(x) - \Phi(y)\|_{\mathcal{H}} > L\|x-y\|$. Take

$$f = \frac{\Phi(x) - \Phi(y)}{\|\Phi(x) - \Phi(y)\|_{\mathcal{H}}} \in \mathcal{H}.$$

We have $\|f\|_{\mathcal{H}} = 1$ and $f(x) - f(y) = \|\Phi(x) - \Phi(y)\|_{\mathcal{H}} > L\|x-y\|$, so that the Lipschitz constant of f is larger than L. □

4.C.2 Proof of Proposition 4.4 (smoothness of 2-layer ReLU NTK)

Proof. Denoting $u = \langle x, y \rangle$, we have

$$\frac{\|\varphi_0(x) - \varphi_0(y)\|^2}{\|x - y\|} = \frac{2\kappa_0(1) - 2\kappa_0(u)}{\sqrt{2 - 2u}}.$$

As a function of $u \in [-1, 1]$, this quantity decreases from 1 to $1/2\pi$, and is thus upper bounded by 1, proving the first part.

Note that if u, v are on the sphere and $\alpha \geq 1$, then $\|u - \alpha v\| \geq \|u - v\|$. This yields

$$\|x - y\| \geq \min(\|x\|, \|y\|)\|\bar{x} - \bar{y}\|,$$

where $\bar{x}, \bar{y}$ denote the normalized vectors. Then, noting that φ_0 is 0-homogeneous, we have

$$\begin{aligned}
\frac{\|\varphi_0(x) - \varphi_0(y)\|^2}{\|x - y\|} &= \frac{\|\varphi_0(\bar{x}) - \varphi_0(\bar{y})\|^2}{\|x - y\|} \\
&\leq \frac{\|\varphi_0(\bar{x}) - \varphi_0(\bar{y})\|^2}{\min(\|x\|, \|y\|)\|\bar{x} - \bar{y}\|} \\
&\leq \frac{1}{\min(\|x\|, \|y\|)},
\end{aligned}$$

by using the previous result on the sphere, and the result for the second part follows.

For the last part, assume x has smaller norm than y. We have

$$\begin{aligned}
\|\Phi(x) - \Phi(y)\| &= \left\| \begin{pmatrix} \varphi_0(x) \otimes x \\ \varphi_1(x) \end{pmatrix} - \begin{pmatrix} \varphi_0(y) \otimes y \\ \varphi_1(y) \end{pmatrix} \right\| \\
&\leq \left\| \begin{pmatrix} \varphi_0(x) \otimes x \\ \varphi_1(x) \end{pmatrix} - \begin{pmatrix} \varphi_0(y) \otimes x \\ \varphi_1(y) \end{pmatrix} \right\| + \left\| \begin{pmatrix} \varphi_0(y) \otimes x \\ \varphi_1(y) \end{pmatrix} - \begin{pmatrix} \varphi_0(y) \otimes y \\ \varphi_1(y) \end{pmatrix} \right\| \\
&= \sqrt{\|x\|^2 \|\varphi_0(x) - \varphi_0(y)\|^2 + \|\varphi_1(x) - \varphi_1(y)\|^2} + \|\varphi_0(y)\|\|x - y\| \\
&\leq \sqrt{\|x\|\|x - y\| + \|x - y\|^2} + \|x - y\| \leq \sqrt{\|x\|\|x - y\|} + 2\|x - y\|,
\end{aligned}$$

where in the last line we used $\|\varphi_0(y)\| = 1$, $\|\varphi_0(x) - \varphi_0(y)\|^2 \leq \|x - y\|/\|x\|$, as well as $\|\varphi_1(x) - \varphi_1(y)\| \leq \|x - y\|$, which follows from Lemma 2.1 in Chapter 2. We conclude by symmetry. □

4.C.3 Proof of Proposition 4.11 (smooth activations)

Proof. We introduce the following kernels defined on the sphere:

$$\kappa_j(\langle x, x' \rangle) = \mathbb{E}_{w \sim \mathcal{N}(0,I)}[\sigma^{(j)}(\langle w, x \rangle)\sigma^{(j)}(\langle w, x' \rangle)].$$

Note that these are indeed dot-product kernels, defined as polynomial expansions in terms of the squared Hermite expansion coefficients of $\sigma^{(j)}$, as shown by Daniely et al. (2016) (called "dual activations"). In fact, Daniely et al. (2016, Lemma 11) also shows that the mapping from activation to dual activation commutes with differentiation, so

that $\kappa_j(u) = \kappa_0^{(j)}(u)$, for $u \in (-1, 1)$. The assumption made in this proposition implies that the j-th order derivatives of κ_0 as $u \to 1$ exist, with $\kappa_0^{(j)}(1) = \kappa_j(1) = \gamma_j < +\infty$.

Then, the NTK on the sphere takes the form $K_\sigma(x, x') = \kappa_\sigma(\langle x, x' \rangle)$, where $\kappa_\sigma(u) = u\kappa_1(u) + \kappa_0(u)$. Then, if we consider the kernel $\hat{\kappa}_\sigma(u) = \frac{\kappa_\sigma(u)}{\kappa_\sigma(1)} = \frac{\kappa_\sigma(u)}{\gamma_0+\gamma_1}$, we have

$$\hat{\kappa}_\sigma(1) = 1 \quad \text{and} \quad \hat{\kappa}'_\sigma(1) = \frac{\kappa_1(1) + \kappa'_1(1) + \kappa'_0(1)}{\gamma_0 + \gamma_1} = \frac{\gamma_2 + 2\gamma_1}{\gamma_0 + \gamma_1}.$$

Applying Lemma 2.1 to this kernel, and re-multiplying by $\gamma_0 + \gamma_1$ yields the final result. □

4.C.4 Proof of Lemma 4.12 (smoothness of operator M in $L^2(\mathbb{R}^d)$)

Proof. Using similar arguments as in the proof of Proposition 4.4, we can show that for any $u \in \mathbb{R}^d$

$$\begin{aligned}\|M(x, y)(u) - M(x', y')(u)\| \leq \sqrt{\min(\|y(u)\|, \|y'(u)\|)\|x(u) - x'(u)\|} \\ + \|x(u) - x'(u)\| + \|y(u) - y'(u)\|\end{aligned}$$

Now assume that $\min(\|y\|, \|y'\|) = \|y\|$. By the triangle inequality in $L^2(\mathbb{R}^d)$, we then have

$$\begin{aligned}\|M(x, y) - M(x', y')\| &\leq \sqrt{\int \min(\|y(u)\|, \|y'(u)\|)\|x(u) - x'(u)\| du} + \|x - x'\| + \|y - y'\| \\ &\leq \sqrt{\int \|y(u)\|\|x(u) - x'(u)\| du} + \|x - x'\| + \|y - y'\| \\ &\leq \sqrt{\|y\|\|x - x'\|} + \|x - x'\| + \|y - y'\|,\end{aligned}$$

where the last inequality follows from Cauchy-Schwarz. We obtain the final result by symmetry. □

4.C.5 Proof of Proposition 4.14 (stability to deformations)

We first recall the following results from Chapter 2.

Lemma 4.17 (Chapter 2, Section 2.3.1)**.** *Assume* $\|\nabla\tau\|_\infty \leq 1/2$*, and* $\sup_{v \in S_k} |v| \leq \beta\sigma_{k-1}$ *for all* k*. We have*

$$\begin{aligned}\|[P^k A^{k-1}, L_\tau]\| &\leq C(\beta)\|\nabla\tau\|_\infty \\ \|L^\tau A_n - A_n\| &\leq \frac{C_2}{\sigma_n}\|\tau\|_\infty,\end{aligned}$$

where $C(\beta)$ *grows with* β *as* β^{d+1}*.*

We are now ready to prove Proposition 4.14.

Proof. In order to compare $\Phi_n(L_\tau x)$ and $\Phi_n(x)$, we introduce intermediate sequences of feature maps, denoted $x_k^{(k_0)}$ and $y_k^{(k_0)}$, where the deformation operator L_τ acts at layer k_0. In particular, we denote by $x_k^{(0)}$, $y_k^{(0)}$ the feature maps obtained for the input $L_\tau x$, and if $k_0 \geq 1$, we define $x_k^{(k_0)} = x_k$, $y_k^{(k_0)} = y_k$ for $k \leq k_0$,

$$\begin{aligned} x_{k_0+1}^{(k_0)} &= P^{k_0+1} A^{k_0} L_\tau \varphi_1(x_{k_0}) \\ y_{k_0+1}^{(k_0)} &= P^{k_0+1} A^{k_0} L_\tau M(x_{k_0}, y_{k_0}), \end{aligned}$$

and for $k \geq k_0 + 2$,

$$\begin{aligned} x_k^{(k_0)} &= P^k A^{k-1} L_\tau \varphi_1(x_{k-1}^{k_0}) \\ y_k^{(k_0)} &= P^k A^{k-1} L_\tau M(x_{k-1}^{k_0}, y_{k-1}^{k_0}). \end{aligned}$$

Then, we have the following

$$\begin{aligned} \|\Phi_n(L_\tau x) - \Phi_n(x)\| &= \|A^n M(x_n^{(0)}, y_n^{(0)}) - A^n M(x_n, y_n)\| \\ &\leq \|A^n M(x_n^{(0)}, y_n^{(0)}) - A^n L_\tau M(x_n, y_n)\| \\ &\quad + \|A^n L_\tau M(x_n, y_n) - A^n M(x_n, y_n)\| \\ &\leq \|A^n M(x_n^{(0)}, y_n^{(0)}) - A^n L_\tau M(x_n, y_n)\| \\ &\quad + \|A^n L_\tau - A^n\| \|M(x_n, y_n)\|. \end{aligned}$$

Using Lemma 4.17, we have

$$\begin{aligned} \|A^n L_\tau - A^n\| &\leq \|[A^n, L_\tau]\| + \|L_\tau A^n - A^n\| \\ &\leq C(\beta)\|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n}\|\tau\|_\infty. \end{aligned}$$

Separately, we have $\|M(x_n, y_n)\|^2 = \|x_n\|^2 + \|y_n\|^2 \leq (n+1)\|x\|^2$, so that

$$\begin{aligned} \|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \|A^n M(x_n^{(0)}, y_n^{(0)}) - A^n L_\tau M(x_n, y_n)\| \\ + \sqrt{n+1}\left(C(\beta)\|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n}\|\tau\|_\infty\right)\|x\|. \end{aligned}$$

We now bound the first term above by induction. For $n = 1$, we have

$$\begin{aligned} \|A^1 M(x_1^{(0)}, y_1^{(0)}) - A^1 L_\tau M(x_1, y_1)\| &\leq \|A^1 M(x_1^{(0)}, y_1^{(0)}) - A^1 L_\tau M(x_1, y_1)\| \\ &\leq \|M(x_1^{(0)}, y_1^{(0)}) - L_\tau M(x_1, y_1)\| \\ &= \|M(L_\tau x, L_\tau x) - L_\tau M(x, y)\| = 0, \end{aligned}$$

by noting that M is a point-wise operator, and thus commutes with L_τ. We now assume $n \geq 2$. We have

$$\begin{aligned} \|A^n M(x_n^{(0)}, y_n^{(0)}) - A^n L_\tau M(x_n, y_n)\| &\leq \|M(x_n^{(0)}, y_n^{(0)}) - L_\tau M(x_n, y_n)\| \\ &= \|M(x_n^{(0)}, y_n^{(0)}) - M(L_\tau x_n, L_\tau y_n)\| \\ &\leq \sqrt{\|L_\tau y_n\| \|x_n^{(0)} - L_\tau x_n\|} \\ &\quad + \|x_n^{(0)} - L_\tau x_n\| + \|y_n^{(0)} - L_\tau y_n\|, \end{aligned}$$

where we used Lemma 4.12 and the fact that M commutes with L_τ.

Now note that since $\|\nabla\tau\|_\infty \le 1/2$, for any signal x we have

$$\begin{aligned}\|L_\tau x\|^2 &= \int \|x(u-\tau(u))\|^2 du = \int \|x(u)\|^2 |\det(I-\nabla\tau(u))|^{-1} du \\ &\le \frac{1}{(1-\|\nabla\tau\|_\infty)^d}\|x\|^2 \le 2^d\|x\|^2. \end{aligned} \tag{4.24}$$

Thus, we have $\|L_\tau y_n\| \le 2^{d/2}\|y_n\| \le \sqrt{2^d n}\|x\|$. Separately, using the non-expansivity of φ_1, we have

$$\begin{aligned}\|x_n^{(0)} - L_\tau x_n\| &\le \sum_{k=1}^{n-1} \|x_n^{(k-1)} - x_n^{(k)}\| + \|x_n^{(n-1)} - L_\tau x_n\| \\ &\le \sum_{k=1}^{n} \|x_k^{(k-1)} - L_\tau x_k\| \\ &= \|P^1 A^0 L_\tau x - L_\tau P^1 A^0 x\| \\ &\quad + \sum_{k=2}^{n} \|P^k A^{k-1} L_\tau \varphi_1(x_{k-1}) - L_\tau P^k A^{k-1}\varphi_1(x_{k-1})\| \\ &\le \sum_{k=1}^{n} \|[P^k A^{k-1}, L_\tau]\|\|x\| \\ &\le C(\beta) n \|\nabla\tau\|_\infty \|x\|, \end{aligned}$$

by Lemma 4.17. We also have

$$\begin{aligned}\|y_n^{(0)} - L_\tau y_n\| &= \|P^n A^{n-1} M(x_{n-1}^{(0)}, y_{n-1}^{(0)}) - L_\tau P^n A^{n-1} M(x_{n-1}, y_{n-1})\| \\ &\le \|P^n A^{n-1} M(x_{n-1}^{(0)}, y_{n-1}^{(0)}) - P^n A^{n-1} L_\tau M(x_{n-1}, y_{n-1})\| \\ &\quad + \|[P^n A^{n-1}, L_\tau]\|\|M(x_{n-1}, y_{n-1})\| \\ &\le \|A^{n-1} M(x_{n-1}^{(0)}, y_{n-1}^{(0)}) - A^{n-1} L_\tau M(x_{n-1}, y_{n-1})\| + C(\beta)\sqrt{n}\|\nabla\tau\|_\infty\|x\|. \end{aligned}$$

We have thus shown:

$$\begin{aligned}&\|A^n M(x_n^{(0)}, y_n^{(0)}) - A^n L_\tau M(x_n, y_n)\| \\ &\qquad \le \left(2^{d/4} C(\beta)^{1/2} n^{3/4} \|\nabla\tau\|_\infty^{1/2} + C(\beta)(n+\sqrt{n})\|\nabla\tau\|_\infty\right)\|x\| \\ &\qquad\quad + \|A^{n-1} M(x_{n-1}^{(0)}, y_{n-1}^{(0)}) - A^{n-1} L_\tau M(x_{n-1}, y_{n-1})\|. \end{aligned}$$

Unrolling the recurrence relation yields

$$\begin{aligned}&\|A^n M(x_n^{(0)}, y_n^{(0)}) - A^n L_\tau M(x_n, y_n)\| \\ &\qquad \le \sum_{k=2}^{n} \left(2^{d/4} C(\beta)^{1/2} k^{3/4} \|\nabla\tau\|_\infty^{1/2} + C(\beta)(k+\sqrt{k})\|\nabla\tau\|_\infty\right)\|x\| \\ &\qquad \le \left(C(\beta)^{1/2} C n^{7/4} \|\nabla\tau\|_\infty^{1/2} + C(\beta) C' n^2 \|\nabla\tau\|_\infty\right)\|x\|, \end{aligned}$$

where C, C' are absolute constants depending only on d.

The final bound becomes

$$\begin{aligned}\|\Phi_n(L_\tau x) - \Phi_n(x)\| \\ \leq \left(C(\beta)^{1/2} C n^{7/4} \|\nabla\tau\|_\infty^{1/2} + C(\beta) C' n^2 \|\nabla\tau\|_\infty + \sqrt{n+1}\frac{C_2}{\sigma_n}\|\tau\|_\infty \right) \|x\|,\end{aligned}$$

with a different constant C'.

□

Chapter 5

Invariance and Stability through Regularization: A Stochastic Optimization Algorithm for Data Augmentation

Stochastic optimization algorithms with variance reduction have proven successful for minimizing large finite sums of functions. Unfortunately, these techniques are unable to deal with stochastic perturbations of input data, induced for example by data augmentation. In such cases, the objective is no longer a finite sum, and the main candidate for optimization is the stochastic gradient descent method (SGD). In this chapter, we introduce a variance reduction approach for these settings when the objective is composite and strongly convex. The convergence rate outperforms SGD with a typically much smaller constant factor, which depends on the variance of gradient estimates only due to perturbations on a *single* example.

This chapter is based on the following paper:

A. Bietti and J. Mairal. Stochastic optimization with variance reduction for infinite datasets with finite sum structure. In *Advances in Neural Information Processing Systems (NIPS)*, 2017b

5.1 Introduction

Many supervised machine learning problems can be cast as the minimization of an expected loss over a data distribution with respect to a vector x in $\mathbb{R}^p$ of model parameters. When an infinite amount of data is available, stochastic optimization methods such as SGD or stochastic mirror descent algorithms, or their variants, are typically used (see Bottou et al., 2018; Duchi and Singer, 2009; Nemirovski et al., 2009; Xiao, 2010). Nevertheless, when the dataset is finite, incremental methods based on variance reduction techniques (*e.g.*, Allen-Zhu, 2017; Defazio et al., 2014a; Johnson and Zhang, 2013; Lan and Zhou, 2017; Lin et al., 2015; Schmidt et al., 2017; Shalev-Shwartz and Zhang, 2013)

have proven to be significantly faster at solving the finite-sum problem

$$\min_{x\in\mathbb{R}^p} \Big\{ F(x) := f(x) + h(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) + h(x) \Big\}, \tag{5.1}$$

where the functions f_i are smooth and convex, and h is a simple convex penalty that need not be differentiable such as the ℓ_1 norm. A classical setting is $f_i(x) = \ell(y_i, x^\top \xi_i) + (\mu/2)\|x\|^2$, where (ξ_i, y_i) is an example-label pair, ℓ is a convex loss function, and μ is a regularization parameter.

In this chapter, we are interested in a variant of (5.1) where random perturbations of data are introduced, which is a common scenario in machine learning. Then, the functions f_i involve an expectation over a random perturbation ρ, leading to the problem

$$\min_{x\in\mathbb{R}^p} \Big\{ F(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + h(x) \Big\}. \quad \text{with} \quad f_i(x) = \mathbb{E}_\rho[\tilde{f}_i(x,\rho)]. \tag{5.2}$$

Unfortunately, variance reduction methods are not compatible with the setting (5.2), since evaluating a single gradient $\nabla f_i(x)$ requires computing a full expectation. Yet, dealing with random perturbations is of utmost interest; for instance, this is a key to achieve stable feature selection (Meinshausen and Bühlmann, 2010), improving the generalization error both in theory (Wager et al., 2014) and in practice (Loosli et al., 2007; van der Maaten et al., 2013), obtaining stable and robust predictors (Zheng et al., 2016), or using complex a priori knowledge about data to generate virtually larger datasets (Loosli et al., 2007; Paulin et al., 2014; Simard et al., 1998). Injecting noise in data is also useful to hide gradient information for privacy-aware learning (Duchi et al., 2012).

Despite its importance, the optimization problem (5.2) has been littled studied and to the best of our knowledge, no dedicated optimization method that is able to exploit the problem structure has been developed so far. A natural way to optimize this objective when $h = 0$ is indeed SGD, but ignoring the finite-sum structure leads to gradient estimates with high variance and slow convergence. The goal of this chapter is to introduce an algorithm for strongly convex objectives, called *stochastic MISO*, which exploits the underlying finite sum using variance reduction. Our method achieves a faster convergence rate than SGD, by removing the dependence on the gradient variance due to sampling the data points i in $\{1, \dots, n\}$; the dependence remains only for the variance due to random perturbations ρ.

To the best of our knowledge, our method is the first algorithm that interpolates naturally between incremental methods for finite sums (when there are no perturbations) and the stochastic approximation setting (when $n = 1$), while being able to efficiently tackle the hybrid case.

Related work. Many optimization methods dedicated to the finite-sum problem (*e.g.*, Johnson and Zhang, 2013; Shalev-Shwartz and Zhang, 2013) have been motivated by the fact that their updates can be interpreted as SGD steps with unbiased estimates of the full gradient, but with a variance that decreases as the algorithm approaches the optimum (Johnson and Zhang, 2013); on the other hand, vanilla SGD requires decreasing step-sizes to achieve this reduction of variance, thereby slowing down convergence. Our work aims at extending these techniques to the case where each function in the finite sum can only be accessed via a first-order stochastic oracle.

Table 5.1: Iteration complexity of different methods for solving the objective (5.2) in terms of number of iterations required to find x such that $\mathbb{E}[f(x) - f(x^*)] \leq \epsilon$. The complexity of N-SAGA (Hofmann et al., 2015) matches the first term of S-MISO but is asymptotically biased. Note that we always have the perturbation noise variance σ_p^2 smaller than the total variance σ_{tot}^2 and thus S-MISO improves on SGD both in the first term (linear convergence to a smaller $\bar{\epsilon}$) and in the second (smaller constant in the asymptotic rate). In many application cases, we also have $\sigma_p^2 \ll \sigma_{\text{tot}}^2$ (see main text and Table 5.2).

Method	Asymptotic error	Iteration complexity
SGD	0	$O\left(\frac{L}{\mu}\log\frac{1}{\bar{\epsilon}} + \frac{\sigma_{\text{tot}}^2}{\mu\epsilon}\right)$ with $\bar{\epsilon} = O\left(\frac{\sigma_{\text{tot}}^2}{\mu}\right)$
N-SAGA	$\epsilon_0 = O\left(\frac{\sigma_p^2}{\mu}\right)$	$O\left(\left(n + \frac{L}{\mu}\right)\log\frac{1}{\epsilon}\right)$ with $\epsilon > \epsilon_0$
S-MISO	0	$O\left(\left(n + \frac{L}{\mu}\right)\log\frac{1}{\bar{\epsilon}} + \frac{\sigma_p^2}{\mu\epsilon}\right)$ with $\bar{\epsilon} = O\left(\frac{\sigma_p^2}{\mu}\right)$

Most related to our work, recent methods that use data clustering to accelerate variance reduction techniques (Allen-Zhu et al., 2016; Hofmann et al., 2015) can be seen as tackling a special case of (5.2), where the expectations in f_i are replaced by empirical averages over points in a cluster. While N-SAGA (Hofmann et al., 2015) was originally not designed for the stochastic context we consider, we remark that their method can be applied to (5.2). Their algorithm is however asymptotically biased and does not converge to the optimum. On the other hand, ClusterSVRG (Allen-Zhu et al., 2016) is not biased, but does not support infinite datasets. The method proposed by Achab et al. (2015) uses variance reduction in a setting where gradients are computed approximately, but the algorithm computes a full gradient at every pass, which is not available in our stochastic setting.

Chapter organization. In Section 5.2, we present our algorithm for smooth objectives, and we analyze its convergence in Section 5.3. We present an extension to composite objectives and non-uniform sampling in Section 5.4. Section 5.5 is devoted to empirical results.

5.2 Stochastic MISO Algorithm for Smooth Objectives

In this section, we introduce the *stochastic MISO* approach for smooth objectives ($h = 0$), which relies on the following assumptions:

(A1) **global strong convexity**: f is μ-strongly convex;

(A2) **smoothness**: $\tilde{f}_i(\cdot, \rho)$ is L-smooth for all i and ρ (*i.e.*, with L-Lipschitz gradients).

Note that these assumptions are relaxed in Appendix 5.4 by supporting composite objectives and by exploiting different smoothness parameters L_i on each example, a setting

Table 5.2: Estimated ratio $\sigma_{\text{tot}}^2/\sigma_p^2$, which corresponds to the expected acceleration of S-MISO over SGD. These numbers are based on feature vectors variance, which is closely related to the gradient variance when learning a linear model. ResNet-50 denotes a 50 layer network (He et al., 2016) pre-trained on the ImageNet dataset. For image transformations, the numbers are empirically evaluated from 100 different images, with 100 random perturbations for each image. R_{tot}^2 (respectively, R_{cluster}^2) denotes the average squared distance between pairs of points in the dataset (respectively, in a given cluster), following Hofmann et al. (2015). The settings for unsupervised CKN (Mairal, 2016) and Scattering (Bruna and Mallat, 2013) are described in Section 5.5. More details are given in the main text.

Perturbation type	**Application case**	**Estimated ratio $\sigma_{\text{tot}}^2/\sigma_p^2$**
Direct perturbation of linear model features	Data clustering	$\approx R_{\text{tot}}^2/R_{\text{cluster}}^2$
	Additive Gaussian noise $\mathcal{N}(0, \alpha^2 I)$	$\approx 1 + 1/\alpha^2$
	Dropout with probability δ	$\approx 1 + 1/\delta$
	Feature rescaling by s in $\mathcal{U}(1-w, 1+w)$	$\approx 1 + 3/w^2$
Random image transformations	ResNet-50, color perturbation	21.9
	ResNet-50, rescaling + crop	13.6
	Unsupervised CKN, rescaling + crop	9.6
	Scattering, gamma correction	9.8

where non-uniform sampling of the training points is typically helpful to accelerate convergence (*e.g.*, Xiao and Zhang, 2014).

Complexity results. We now introduce the following quantity, which is essential in our analysis:

$$\sigma_p^2 := \frac{1}{n}\sum_{i=1}^{n} \sigma_i^2, \quad \text{with} \quad \sigma_i^2 := \mathbb{E}_\rho \left[\|\nabla \tilde{f}_i(x^*, \rho) - \nabla f_i(x^*)\|^2 \right],$$

where x^* is the (unique) minimizer of f. The quantity σ_p^2 represents the part of the variance of the gradients at the optimum that is due to the perturbations ρ. In contrast, another quantity of interest is the total variance σ_{tot}^2, which also includes the randomness in the choice of the index i, defined as

$$\sigma_{\text{tot}}^2 = \mathbb{E}_{i,\rho}[\|\nabla \tilde{f}_i(x^*, \rho)\|^2] = \sigma_p^2 + \mathbb{E}_i[\|\nabla f_i(x^*)\|^2] \qquad \text{(note that } \nabla f(x^*) = 0\text{)}.$$

The relation between σ_{tot}^2 and σ_p^2 is obtained by simple algebraic manipulations.

The goal of our work is to exploit the potential imbalance $\sigma_p^2 \ll \sigma_{\text{tot}}^2$, occurring when perturbations on input data are small compared to the sampling noise. The assumption is reasonable: given a data point, selecting a different one should lead to larger variation than a simple perturbation. From a theoretical point of view, the approach we propose achieves the iteration complexity presented in Table 5.1, see also Appendix 5.C and Bach and Moulines (2011); Bottou et al. (2018); Nemirovski et al. (2009) for the complexity analysis of SGD. The gain over SGD is of order $\sigma_{\text{tot}}^2/\sigma_p^2$, which is also observed in our experiments in Section 5.5. We also compare against the method N-SAGA; its convergence rate is similar to ours but suffers from a non-zero asymptotic error.

Algorithm 1 S-MISO for smooth objectives

Input: step-size sequence $(\alpha_t)_{t\geq 1}$;
initialize $x_0 = \frac{1}{n}\sum_i z_i^0$ for some $(z_i^0)_{i=1,\dots,n}$;
for $t = 1, \dots$ **do**
Sample an index i_t uniformly at random, a perturbation ρ_t, and update

$$z_i^t = \begin{cases} (1-\alpha_t) z_i^{t-1} + \alpha_t (x_{t-1} - \frac{1}{\mu} \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t)), & \text{if } i = i_t \\ z_i^{t-1}, & \text{otherwise.} \end{cases} \tag{5.3}$$

$$x_t = \frac{1}{n}\sum_{i=1}^n z_i^t = x_{t-1} + \frac{1}{n}(z_{i_t}^t - z_{i_t}^{t-1}). \tag{5.4}$$

end for

Motivation from application cases. One clear framework of application is the data clustering scenario already investigated by Allen-Zhu et al. (2016); Hofmann et al. (2015). Nevertheless, we will focus on less-studied data augmentation settings that lead instead to true stochastic formulations such as (5.2). First, we consider learning a linear model when adding simple direct manipulations of feature vectors, via rescaling (multiplying each entry vector by a random scalar), Dropout, or additive Gaussian noise, in order to improve the generalization error (Wager et al., 2014) or to get more stable estimators (Meinshausen and Bühlmann, 2010). In Table 5.2, we present the potential gain over SGD in these scenarios. To do that, we study the variance of perturbations applied to a feature vector ξ. Indeed, the gradient of the loss is proportional to ξ, which allows us to obtain good estimates of the ratio $\sigma_{\text{tot}}^2/\sigma_p^2$, as we observed in our empirical study of Dropout presented in Section 5.5. Whereas some perturbations are friendly for our method such as feature rescaling (a rescaling window of $[0.9, 1.1]$ yields for instance a huge gain factor of 300), a large Dropout rate would lead to less impressive acceleration (*e.g.*, a Dropout with $\delta = 0.5$ simply yields a factor 2).

Second, we also consider more interesting domain-driven data perturbations such as classical image transformations considered in computer vision (Paulin et al., 2014; Zheng et al., 2016) including image cropping, rescaling, brightness, contrast, hue, and saturation changes. These transformations may be used to train a linear classifier on top of an unsupervised multilayer image model such as unsupervised CKNs (Mairal, 2016) or the scattering transform (Bruna and Mallat, 2013). It may also be used for retraining the last layer of a pre-trained deep neural network: given a new task unseen during the full network training and given limited amount of training data, data augmentation may be indeed crucial to obtain good prediction and S-MISO can help accelerate learning in this setting. These scenarios are also studied in Table 5.2, where the experiment with ResNet-50 involving random cropping and rescaling produces 224×224 images from 256×256 ones. For these scenarios with realistic perturbations, the potential gain varies from 10 to 20.

Description of stochastic MISO. We are now in shape to present our method, described in Algorithm 1. Without perturbations and with a constant step-size, the algorithm resembles the MISO/Finito algorithms (Defazio et al., 2014b; Lin et al., 2015;

Mairal, 2015), which may be seen as primal variants of SDCA (Shalev-Shwartz, 2016; Shalev-Shwartz and Zhang, 2013). Specifically, MISO is not able to deal with our stochastic objective (5.2), but it may address the deterministic finite-sum problem (5.1). It is part of a larger body of optimization methods that iteratively build a *model* of the objective function, typically a lower or upper bound on the objective that is easier to optimize; for instance, this strategy is commonly adopted in bundle methods (Hiriart-Urruty and Lemaréchal, 1993; Nesterov, 2004).

More precisely, MISO assumes that each f_i is strongly convex and builds a model using lower bounds $D_t(x) = \frac{1}{n}\sum_{i=1}^n d_i^t(x)$, where each d_i^t is a quadratic lower bound on f_i of the form

$$d_i^t(x) = c_{i,1}^t + \frac{\mu}{2}\|x - z_i^t\|^2 = c_{i,2}^t - \mu\langle x, z_i^t\rangle + \frac{\mu}{2}\|x\|^2. \tag{5.5}$$

These lower bounds are updated during the algorithm using strong convexity lower bounds at x_{t-1} of the form $l_i^t(x) = f_i(x_{t-1}) + \langle \nabla f_i(x_{t-1}), x - x_{t-1}\rangle + \frac{\mu}{2}\|x - x_{t-1}\|^2 \leq f_i(x)$:

$$d_i^t(x) = \begin{cases} (1-\alpha_t) d_i^{t-1}(x) + \alpha_t l_i^t(x), & \text{if } i = i_t \\ d_i^{t-1}(x), & \text{otherwise,} \end{cases} \tag{5.6}$$

which corresponds to an update of the quantity z_i^t:

$$z_i^t = \begin{cases} (1-\alpha_t) z_i^{t-1} + \alpha_t (x_{t-1} - \frac{1}{\mu}\nabla f_{i_t}(x_{t-1})), & \text{if } i = i_t \\ z_i^{t-1}, & \text{otherwise.} \end{cases}$$

The next iterate is then computed as $x_t = \arg\min_x D_t(x)$, which is equivalent to (5.4). The original MISO/Finito algorithms use $\alpha_t = 1$ under a "big data" condition on the sample size n (Defazio et al., 2014b; Mairal, 2015), while the theory was later extended by Lin et al. (2015) to relax this condition by supporting smaller constant steps $\alpha_t = \alpha$, leading to an algorithm that may be interpreted as a primal variant of SDCA (see Shalev-Shwartz, 2016).

Note that when f_i is an expectation, it is hard to obtain such lower bounds since the gradient $\nabla f_i(x_{t-1})$ is not available in general. For this reason, we have introduced S-MISO, which can exploit *approximate* lower bounds to each f_i using gradient estimates, by letting the step-sizes α_t decrease appropriately as commonly done in stochastic approximation. This leads to update (5.3).

Separately, SDCA (Shalev-Shwartz and Zhang, 2013) considers the Fenchel conjugates of f_i, defined by $f_i^*(y) = \sup_x x^\top y - f_i(x)$. When f_i is an expectation, f_i^* is not available in closed form in general, nor are its gradients, and in fact exploiting stochastic gradient estimates is difficult in the duality framework. In contrast, Shalev-Shwartz (2016) gives an analysis of SDCA in the primal, aka. "without duality", for smooth finite sums, and our work extends this line of reasoning to the stochastic approximation and composite settings.

Relationship with SGD in the smooth case. The link between S-MISO in the non-composite setting and SGD can be seen by rewriting the update (5.4) as

$$x_t = x_{t-1} + \frac{1}{n}(z_{i_t}^t - z_{i_t}^{t-1}) = x_{t-1} + \frac{\alpha_t}{n} v_t,$$

where

$$v_t := x_{t-1} - \frac{1}{\mu}\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - z_{i_t}^{t-1}. \tag{5.7}$$

Note that $\mathbb{E}[v_t|\mathcal{F}_{t-1}] = -\frac{1}{\mu}\nabla f(x_{t-1})$, where $\mathcal{F}_{t-1}$ contains all information up to iteration t; hence, the algorithm can be seen as an instance of the stochastic gradient method with unbiased gradients, which was a key motivation in SVRG Johnson and Zhang (2013) and later in other variance reduction algorithms Defazio et al. (2014a); Shalev-Shwartz (2016). It is also worth noting that in the absence of a finite-sum structure ($n=1$), we have $z_{i_t}^{t-1} = x_{t-1}$; hence our method becomes identical to SGD, up to a redefinition of step-sizes. In the composite case (see Section 5.4), our approach yields a new algorithm that resembles regularized dual averaging (Xiao, 2010).

Memory requirements and handling of sparse datasets. The algorithm requires storing the vectors $(z_i^t)_{i=1,\dots,n}$, which takes the same amount of memory as the original dataset and which is therefore a reasonable requirement in many practical cases. In the case of sparse datasets, it is fair to assume that random perturbations applied to input data preserve the sparsity patterns of the original vectors, as is the case, *e.g.*, when applying Dropout to text documents described with bag-of-words representations (Wager et al., 2014). If we further assume the typical setting where the μ-strong convexity comes from an ℓ_2 regularizer: $\tilde{f}_i(x,\rho) = \phi_i(x^\top \xi_i^\rho) + (\mu/2)\|x\|^2$, where ξ_i^ρ is the (sparse) perturbed example and ϕ_i encodes the loss, then the update (5.3) can be written as

$$z_i^t = \begin{cases} (1-\alpha_t) z_i^{t-1} - \frac{\alpha_t}{\mu}\phi_i'(x_{t-1}^\top \xi_i^{\rho_t})\xi_i^{\rho_t}, & \text{if } i = i_t \\ z_i^{t-1}, & \text{otherwise,} \end{cases}$$

which shows that for every index i, the vector z_i^t preserves the same sparsity pattern as the examples ξ_i^ρ throughout the algorithm (assuming the initialization $z_i^0 = 0$), making the update (5.3) efficient. The update (5.4) has the same cost since $v_t = z_{i_t}^t - z_{i_t}^{t-1}$ is also sparse.

Limitations and subsequent work. Since our algorithm is uniformly better than SGD in terms of iteration complexity, its main limitation is in terms of memory storage when the dataset cannot fit into memory (remember that the memory cost of S-MISO is the same as the input dataset). In these huge-scale settings, SGD should be preferred; this holds true in fact for all incremental methods when one cannot afford to perform more than one (or very few) passes over the data. Our work focuses instead on non-huge datasets, which are those benefiting most from data augmentation.

We note that a different approach to variance reduction like SVRG (Johnson and Zhang, 2013) is able to trade off storage requirements for additional full gradient computations, which would be desirable in some situations. In particular, previous papers considered constant step-size approaches based for similar settings with improved storage requirements by relying on an gradient estimator similar to SVRG (Achab et al., 2015; Hofmann et al., 2015), however these either maintain a non-zero asymptotic error, or require dynamically reducing the variance of gradient estimates. After our work was published, other papers tackled a similar setting to ours, with improved storage requirements (Zheng and Kwok, 2018; Kulunchakov and Mairal, 2019). The method of Zheng

and Kwok (2018) considers a variant of SAGA for a setting with stochastic perturbations, as in our case, but where the average perturbed data is known (*e.g.*, for Dropout noise, the average perturbed example is just the original example), which enables storing scalars instead of vectors. Kulunchakov and Mairal (2019) present a method similar to SVRG that works in a setting like ours, but overcomes storage requirements by trading off storage for computation as in SVRG.

5.3 Convergence Analysis of S-MISO

We now study the convergence properties of the S-MISO algorithm. For space limitation reasons, all proofs are provided in Appendix 5.A. We start by defining the problem-dependent quantities $z_i^* := x^* - \frac{1}{\mu}\nabla f_i(x^*)$, and then introduce the Lyapunov function

$$C_t = \frac{1}{2}\|x_t - x^*\|^2 + \frac{\alpha_t}{n^2}\sum_{i=1}^{n}\|z_i^t - z_i^*\|^2. \tag{5.8}$$

Proposition 5.1 gives a recursion on C_t, obtained by upper-bounding separately its two terms, and finding coefficients to cancel out other appearing quantities when relating C_t to C_{t-1}. To this end, we borrow elements of the convergence proof of SDCA without duality (Shalev-Shwartz, 2016); our technical contribution is to extend their result to the stochastic approximation and composite (see Section 5.4) cases.

Proposition 5.1 (Recursion on C_t)**.** *If $(\alpha_t)_{t\geq 1}$ is a positive and non-increasing sequence satisfying*

$$\alpha_1 \leq \min\left\{\frac{1}{2}, \frac{n}{2(2\kappa - 1)}\right\}, \tag{5.9}$$

with $\kappa = L/\mu$, then C_t obeys the recursion

$$\mathbb{E}[C_t] \leq \left(1 - \frac{\alpha_t}{n}\right)\mathbb{E}[C_{t-1}] + 2\left(\frac{\alpha_t}{n}\right)^2\frac{\sigma_p^2}{\mu^2}. \tag{5.10}$$

We now state the main convergence result, which provides the expected rate $O(1/t)$ on C_t based on decreasing step-sizes, similar to Bottou et al. (2018) for SGD. Note that convergence of objective function values is directly related to that of the Lyapunov function C_t via smoothness:

$$\mathbb{E}[f(x_t) - f(x^*)] \leq \frac{L}{2}\,\mathbb{E}\left[\|x_t - x^*\|^2\right] \leq L\,\mathbb{E}[C_t]. \tag{5.11}$$

Theorem 5.2 (Convergence of Lyapunov function)**.** *Let the sequence of step-sizes $(\alpha_t)_{t\geq 1}$ be defined by $\alpha_t = \frac{2n}{\gamma+t}$ with $\gamma \geq 0$ such that α_1 satisfies* (5.9)*. For all $t \geq 0$, it holds that*

$$\mathbb{E}[C_t] \leq \frac{\nu}{\gamma + t + 1} \quad \text{where} \quad \nu := \max\left\{\frac{8\sigma_p^2}{\mu^2}, (\gamma+1)C_0\right\}. \tag{5.12}$$

Choice of step-sizes in practice. Naturally, we would like ν to be small, in particular independent of the initial condition C_0 and equal to the first term in the definition (5.12). We would like the dependence on C_0 to vanish at a faster rate than $O(1/t)$, as it is the case in variance reduction algorithms on finite sums. As advised by Bottou et al. (2018) in the context of SGD, we can initially run the algorithm with a constant step-size $\bar{\alpha}$ and exploit this linear convergence regime until we reach the level of noise given by σ_p, and then start decaying the step-size. It is easy to see that by using a constant step-size $\bar{\alpha}$, C_t converges near a value $\bar{C} := 2\bar{\alpha}\sigma_p^2/n\mu^2$. Indeed, Eq. (5.10) with $\alpha_t = \bar{\alpha}$ yields

$$\mathbb{E}[C_t - \bar{C}] \leq \left(1 - \frac{\bar{\alpha}}{n}\right) \mathbb{E}[C_{t-1} - \bar{C}].$$

Thus, we can reach a precision C_0' with $\mathbb{E}[C_0'] \leq \bar{\epsilon} := 2\bar{C}$ in $O(\frac{n}{\bar{\alpha}} \log C_0/\bar{\epsilon})$ iterations. Then, if we start decaying step-sizes as in Theorem 5.2 with γ large enough so that $\alpha_1 = \bar{\alpha}$, we have

$$(\gamma + 1)\, \mathbb{E}[C_0'] \leq (\gamma + 1)\bar{\epsilon} = 8\sigma_p^2/\mu^2,$$

making both terms in (5.12) smaller than or equal to $\nu = 8\sigma_p^2/\mu^2$. Considering these two phases, with an initial step-size $\bar{\alpha}$ given by (5.9), the final work complexity for reaching $\mathbb{E}[\|x_t - x^*\|^2] \leq \epsilon$ is

$$O\left(\left(n + \frac{L}{\mu}\right) \log \frac{C_0}{\bar{\epsilon}}\right) + O\left(\frac{\sigma_p^2}{\mu^2 \epsilon}\right). \tag{5.13}$$

We can then use (5.11) in order to obtain the complexity for reaching $\mathbb{E}[f(x_t) - f(x^*)] \leq \epsilon$. Note that following this step-size strategy was found to be very effective in practice (see Section 5.5).

Acceleration by iterate averaging. When one is interested in the convergence in function values, the complexity (5.13) combined with (5.11) yields $O(L\sigma_p^2/\mu^2\epsilon)$, which can be problematic for ill-conditioned problems (large condition number L/μ). The following theorem presents an iterate averaging scheme which brings the complexity term down to $O(\sigma_p^2/\mu\epsilon)$, which appeared in Table 5.1.

Theorem 5.3 (Convergence under iterate averaging)**.** *Let the step-size sequence* $(\alpha_t)_{t\geq 1}$ *be defined by*

$$\alpha_t = \frac{2n}{\gamma + t} \quad \text{for } \gamma \geq 1 \text{ s.t. } \alpha_1 \leq \min\left\{\frac{1}{2}, \frac{n}{4(2\kappa - 1)}\right\}.$$

We have

$$\mathbb{E}[f(\bar{x}_T) - f(x^*)] \leq \frac{2\mu\gamma(\gamma - 1)C_0}{T(2\gamma + T - 1)} + \frac{16\sigma_p^2}{\mu(2\gamma + T - 1)},$$

where

$$\bar{x}_T := \frac{2}{T(2\gamma + T - 1)} \sum_{t=0}^{T-1} (\gamma + t) x_t.$$

The proof uses a similar telescoping sum technique to Lacoste-Julien et al. (2012). Note that if $T \gg \gamma$, the first term, which depends on the initial condition C_0, decays as $1/T^2$ and is thus dominated by the second term. Moreover, if we start averaging

after an initial phase with constant step-size $\bar{\alpha}$, we can consider $C_0 \approx 4\bar{\alpha}\sigma_p^2/n\mu^2$. In the ill-conditioned regime, taking $\bar{\alpha} = \alpha_1 = 2n/(\gamma+1)$ as large as allowed by (5.9), we have γ of the order of $\kappa = L/\mu \gg 1$. The full convergence rate then becomes

$$\mathbb{E}[f(\bar{x}_T) - f(x^*)] \leq O\left(\frac{\sigma_p^2}{\mu(\gamma+T)}\left(1+\frac{\gamma}{T}\right)\right).$$

When T is large enough compared to γ, this becomes $O(\sigma_p^2/\mu T)$, leading to a complexity $O(\sigma_p^2/\mu\epsilon)$.

Comparison with subsequent work. We note that in subsequent work, Kulunchakov and Mairal (2019) obtained similar or improved convergence rates for more general algorithms, including accelerated algorithms which obtain faster convergence in the initial constant step-size phase, without compromising the dependence of complexity on noise. The S-SAGA method of Zheng and Kwok (2018), which requires knowledge of expected feature vectors under perturbations, obtains similar convergence rates to ours, but with a dependence on a different noise quantity which may be smaller than σ_p in machine learning problems when datapoints and their perturbations have small loss.

5.4 Extension to Composite Objectives and Non-Uniform Sampling

In this section, we study extensions of S-MISO to different situations where our previous smoothness assumption (A2) is not suitable, either because of a non-smooth term h in the objective or because it ignores additional useful knowledge about each f_i such as the norm of each example.

In the presence of non-smooth regularizers such as the ℓ_1-norm, the objective is no longer smooth, but we can leverage its composite structure by using proximal operators. To this end, we assume that one can easily compute the proximal operator of h, defined by

$$\mathrm{prox}_h(z) := \arg\min_{x\in\mathbb{R}^p}\left\{\frac{1}{2}\|x-z\|^2 + h(x)\right\}.$$

When the smoothness constants L_i vary significantly across different examples (typically through the norm of the feature vectors), the uniform upper bound $L = L_{\max} = \max_i L_i$ can be restrictive. It has been noticed (see, *e.g.*, Schmidt et al., 2017; Xiao and Zhang, 2014) that when the L_i are known, one can achieve better convergence rates—typically depending on the average smoothness constant $\bar{L} = \frac{1}{n}\sum_i L_i$ rather than $L_{\max}$—by sampling examples in a non-uniform way. For that purpose, we now make the following assumptions:

- (A3) **strong convexity**: $\tilde{f}_i(\cdot,\rho)$ is μ-strongly convex for all i,ρ;
- (A4) **smoothness**: $\tilde{f}_i(\cdot,\rho)$ is L_i-smooth for all i,ρ;

Note that our proof relies on a slightly stronger assumption (A3) than the global strong convexity assumption (A1) made above, which holds in the situation where strong convexity comes from an ℓ_2 regularization term. In order to exploit the different smoothness constants, we allow the algorithm to sample indices i non-uniformly, from any distribution q such that $q_i \geq 0$ for all i and $\sum_i q_i = 1$.

Algorithm 2 S-MISO for composite objectives, with non-uniform sampling.

Input: step-sizes $(\alpha_t)_{t\geq 1}$, sampling distribution q;
Initialize $x_0 = \mathrm{prox}_{h/\mu}(\bar{z}_0)$ with $\bar{z}_0 = \frac{1}{n}\sum_i z_i^0$ for some $(z_i^0)_{i=1,\dots,n}$ that safisfies (5.16);
for $t = 1, \dots$ **do**
Sample an index $i_t \sim q$, a perturbation ρ_t, and update (with $\alpha_t^i = \frac{\alpha_t}{q_i n}$):

$$z_i^t = \begin{cases} (1-\alpha_t^i) z_i^{t-1} + \alpha_t^i (x_{t-1} - \frac{1}{\mu}\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t)), & \text{if } i = i_t \\ z_i^{t-1}, & \text{otherwise} \end{cases} \tag{5.14}$$

$$\bar{z}_t = \frac{1}{n}\sum_{i=1}^n z_i^t = \bar{z}_{t-1} + \frac{1}{n}(z_{i_t}^t - z_{i_t}^{t-1})$$

$$x_t = \mathrm{prox}_{h/\mu}(\bar{z}_t). \tag{5.15}$$

end for

The extension of S-MISO to this setting is given in Algorithm 2. Note that the step-sizes vary depending on the example, with larger steps for examples that are sampled less frequently (typically "easier" examples with smaller L_i). Note that when $h = 0$, the update directions are unbiased estimates of the gradient: we have $\mathbb{E}[x_t - x_{t-1}|\mathcal{F}_{t-1}] = -\frac{\alpha_t}{n\mu}\nabla f(x_{t-1})$ as in the uniform case. However, in the composite case, the algorithm cannot be written in a proximal stochastic gradient form like Prox-SVRG (Xiao and Zhang, 2014) or SAGA (Defazio et al., 2014a).

Relationship with RDA. When $n = 1$, our algorithm performs similar updates to Regularized Dual Averaging (RDA, Xiao, 2010) with strongly convex regularizers. In particular, if $\tilde{f}_1(x, \rho) = \phi(x^\top \xi(\rho)) + (\mu/2)\|x\|^2$, the updates are the same when taking $\alpha_t = 1/t$, since

$$\mathrm{prox}_{h/\mu}(\bar{z}_t) = \arg\min_x \left\{ \langle -\mu \bar{z}_t, x\rangle + \frac{\mu}{2}\|x\|^2 + h(x) \right\},$$

and $-\mu\bar{z}_t$ is equal to the average of the gradients of the loss term up to t, which appears in the same way in the RDA updates (Xiao, 2010, Section 2.2). However, unlike RDA, our method supports arbitrary decreasing step-sizes, in particular keeping the step-size constant, which can lead to faster convergence in the initial iterations (see Section 5.3).

Lower-bound model and convergence analysis. Again, we can view the algorithm as iteratively updating approximate lower bounds on the objective F of the form $D_t(x) = \frac{1}{n}\sum_i d_i^t(x) + h(x)$ analogously to (5.6), and minimizing the new D_t in (5.15). Similar to MISO-Prox, we require that d_i^0 is initialized with a μ-strongly convex quadratic such that $\tilde{f}_i(x, \tilde{\rho}_i) \geq d_i^0(x)$ with the random perturbation $\tilde{\rho}_i$. Given the form of d_i^t in (5.5), it suffices to choose z_i^0 that satisfies

$$\tilde{f}_i(x, \tilde{\rho}_i) \geq \frac{\mu}{2}\|x - z_i^0\| + c, \tag{5.16}$$

for some constant c. In the common case of an ℓ_2 regularizer with a non-negative loss, one can simply choose $z_i^0 = 0$ for all i, otherwise, z_i^0 can be obtained by considering

a strong convexity lower bound on $\tilde{f}_i(\cdot, \tilde{\rho}_i)$. Our new analysis relies on the minimum $D_t(x_t)$ of the lower bounds D_t through the following Lyapunov function:

$$C_t^q = F(x^*) - D_t(x_t) + \frac{\mu\alpha_t}{n^2} \sum_{i=1}^{n} \frac{1}{q_i n} \|z_i^t - z_i^*\|^2. \tag{5.17}$$

The convergence of the iterates x_t is controlled by the convergence in C_t^q thanks to the next lemma:

Lemma 5.4 (Bound on the iterates)**.** *For all t, we have*

$$\frac{\mu}{2} \mathbb{E}[\|x_t - x^*\|^2] \leq \mathbb{E}[F(x^*) - D_t(x_t)]. \tag{5.18}$$

The following proposition gives a recursion on C_t^q similar to Proposition 5.1.

Proposition 5.5 (Recursion on C_t^q)**.** *If $(\alpha_t)_{t\geq 1}$ is a positive and non-increasing sequence of step-sizes satisfying*

$$\alpha_1 \leq \min\left\{\frac{n q_{\min}}{2}, \frac{n\mu}{4L_q}\right\}, \tag{5.19}$$

with $q_{\min} = \min_i q_i$ and $L_q = \max_i \frac{L_i - \mu}{q_i n}$, then C_t^q obeys the recursion

$$\mathbb{E}[C_t^q] \leq \left(1 - \frac{\alpha_t}{n}\right) \mathbb{E}[C_{t-1}^q] + 2\left(\frac{\alpha_t}{n}\right)^2 \frac{\sigma_q^2}{\mu}, \tag{5.20}$$

with $\sigma_q^2 = \frac{1}{n}\sum_i \frac{\sigma_i^2}{q_i n}$.

Note that if we consider the quantity $\mathbb{E}[C_t^q/\mu]$, which is an upper bound on $\frac{1}{2}\mathbb{E}[\|x_t - x^*\|^2]$ by Lemma 5.4, we have the same recursion as (5.10), and thus can apply Theorem 5.2 with the new condition (5.19). If we choose

$$q_i = \frac{1}{2n} + \frac{L_i - \mu}{2\sum_i (L_i - \mu)}, \tag{5.21}$$

we have $q_{\min} \geq 1/2n$ and $L_q \leq 2(\bar{L} - \mu)$, where $\bar{L} = \frac{1}{n}\sum_i L_i$. Then, taking $\alpha_1 = \min(1/4, n\mu/8(\bar{L} - \mu))$ satisfies (5.19), and using similar arguments to Section 5.3, the complexity for reaching $\mathbb{E}[\|x_t - x^*\|^2] \leq \epsilon$ is

$$O\left(\left(n + \frac{\bar{L}}{\mu}\right) \log \frac{C_0^q}{\bar{\epsilon}}\right) + O\left(\frac{\sigma_q^2}{\mu^2 \epsilon}\right),$$

where $\bar{\epsilon} = 4\bar{\alpha}\sigma_q^2/n\mu$, and $\bar{\alpha}$ is the initial constant step-size. For the complexity in function suboptimality, the second term becomes $O(\sigma_q^2/\mu\epsilon)$ by using the same averaging scheme presented in Theorem 5.3 and adapting the proof. Note that with our choice of q, we have $\sigma_q^2 \leq \frac{2}{n}\sum_i \sigma_i^2 = 2\bar{\sigma_p}^2$, for general perturbations, where $\bar{\sigma_p}^2 = \frac{1}{n}\sum_i \sigma_i^2$ is the variance in the uniform case. Additionally, it is often reasonable to assume that the variance from perturbations increases with the norm of examples, for instance Dropout perturbations get larger when coordinates have larger magnitudes. Based on this observation, if we make the assumption that $\sigma_i^2 \propto L_i - \mu$, that is $\sigma_i^2 = \bar{\sigma_p}^2 \frac{L_i - \mu}{\bar{L} - \mu}$, then for both $q_i = 1/n$ (uniform case) and $q_i = (L_i - \mu)/n(\bar{L} - \mu)$, we have $\sigma_q^2 = \bar{\sigma_p}^2$, and thus we have $\sigma_q^2 \leq \bar{\sigma_p}^2$ for the choice of q given in (5.21), since σ_q^2 is convex in q. Thus, we can expect that the $O(1/t)$ convergence phase behaves similarly or better than for uniform sampling, which is confirmed by our experiments (see Section 5.5).

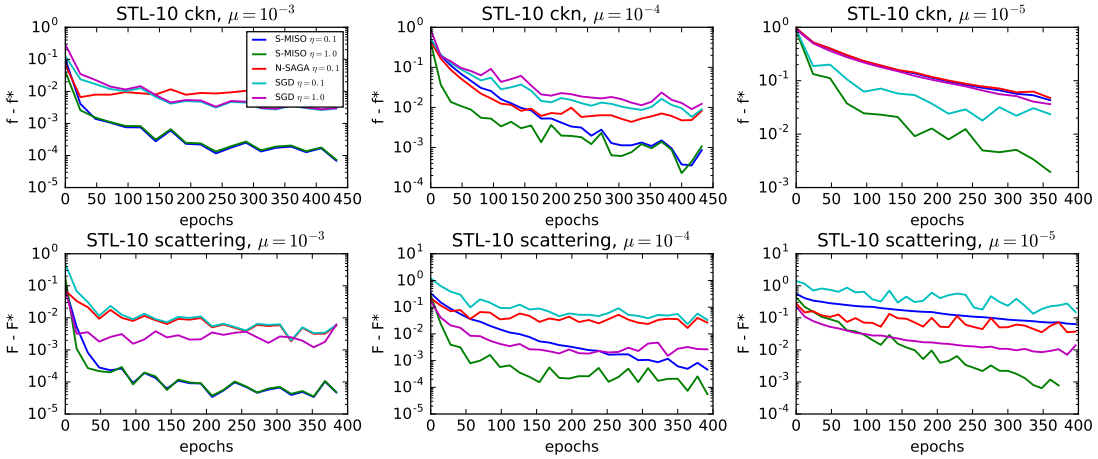


Figure 5.1: Impact of conditioning for data augmentation on STL-10 (controlled by μ, where $\mu = 10^{-4}$ gives the best accuracy). Values of the loss are shown on a **logarithmic scale** (1 unit = factor 10). $\eta = 0.1$ satisfies the theory for all methods, and we include curves for larger step-sizes $\eta = 1$. We omit N-SAGA for $\eta = 1$ because it remains far from the optimum. For the scattering representation, the problem we study is ℓ_1-regularized, and we use the composite algorithm of Appendix 5.4.

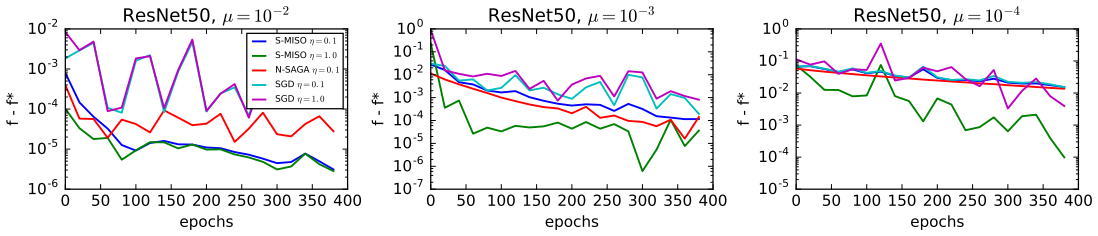


Figure 5.2: Re-training of the last layer of a pre-trained ResNet 50 model, on a small dataset with random color perturbations (for different values of μ).

5.5 Experiments

We present experiments comparing S-MISO with SGD and N-SAGA (Hofmann et al., 2015) on four different scenarios, in order to demonstrate the wide applicability of our method: we consider an image classification dataset with two different image representations and random transformations, and two classification tasks with Dropout regularization, one on genetic data, and one on (sparse) text data. Figures 5.1 and 5.3 show the curves for an estimate of the training objective using 5 sampled perturbations per example. The plots are shown on a logarithmic scale, and the values are compared to the best value obtained among the different methods in 500 epochs. The strong convexity constant μ is the regularization parameter. For all methods, we consider step-sizes supported by the theory as well as larger step-sizes that may work better in practice. Our C++/Cython implementation of all methods considered in this section is available at `https://github.com/albietz/stochs`.

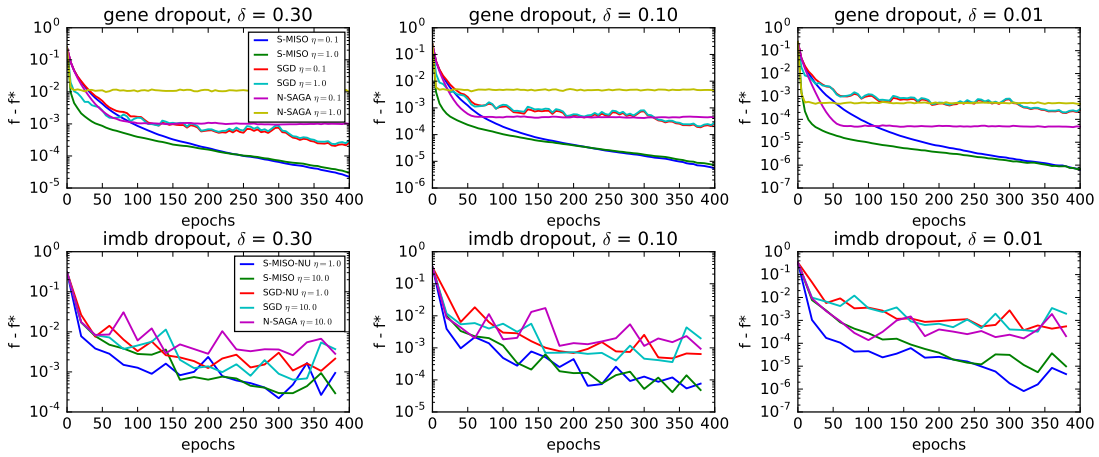


Figure 5.3: Impact of perturbations controlled by the Dropout rate δ. The gene data is ℓ_2-normalized; hence, we consider similar step-sizes as Figure 5.1. The IMDB dataset is highly heterogeneous; thus, we also include non-uniform (NU) sampling variants of Appendix 5.4. For uniform sampling, theoretical step-sizes perform poorly for all methods; thus, we show a larger tuned step-size $\eta = 10$.

Choices of step-sizes. For both S-MISO and SGD, we use the step-size strategy mentioned in Section 5.3 and advised by Bottou et al. (2018), which we have found to be most effective among many heuristics we have tried: we initially keep the step-size constant (controlled by a factor $\eta \leq 1$ in the figures) for 2 epochs, and then start decaying as $\alpha_t = C/(\gamma + t)$, where $C = 2n$ for S-MISO, $C = 2/\mu$ for SGD, and γ is chosen large enough to match the previous constant step-size. For N-SAGA, we maintain a constant step-size throughout the optimization, as suggested in the original paper (Hofmann et al., 2015). The factor η shown in the figures is such that $\eta = 1$ corresponds to an initial step-size $n\mu/(L - \mu)$ for S-MISO (from (5.19) in the uniform case) and $1/L$ for SGD and N-SAGA (with $\bar{L}$ instead of L in the non-uniform case when using the variant of Appendix 5.4).

Image classification with "data augmentation". The success of deep neural networks is often limited by the availability of large amounts of labeled images. When there are many unlabeled images but few labeled ones, a common approach is to train a linear classifier on top of a deep network learned in an unsupervised manner, or pre-trained on a different task (*e.g.*, on the ImageNet dataset). We follow this approach on the STL-10 dataset Coates et al. (2011), which contains 5K training images from 10 classes and 100K unlabeled images, using a 2-layer unsupervised convolutional kernel network Mairal (2016), giving representations of dimension 9 216. The perturbation consists of randomly cropping and scaling the input images. We use the squared hinge loss in a one-versus-all setting. The vector representations are ℓ_2-normalized such that we may use the upper bound $L = 1 + \mu$ for the smoothness constant. We also present results on the same dataset using a scattering representation (Bruna and Mallat, 2013) of dimension 21 696, with random gamma corrections (raising all pixels to the power γ, where γ is chosen randomly around 1). For this representation, we add an ℓ_1 regularization term and use the composite variant of S-MISO presented in Appendix 5.4.

Figure 5.1 shows convergence results on one training fold (500 images), for different values of μ, allowing us to study the behavior of the algorithms for different condition numbers. The low variance induced by data transformations allows S-MISO to reach suboptimality that is orders of magnitude smaller than SGD after the same number of epochs. Note that one unit on these plots corresponds to one order of magnitude in the logarithmic scale. N-SAGA initially reaches a smaller suboptimality than SGD, but quickly gets stuck due to the bias in the algorithm, as predicted by the theory Hofmann et al. (2015), while S-MISO and SGD continue to converge to the optimum thanks to the decreasing step-sizes. The best validation accuracy for both representations is obtained for $\mu \approx 10^{-4}$ (middle column), and we observed relative gains of up to 1% from using data augmentation. We computed empirical variances of the image representations for these two strategies, which are closely related to the variance in gradient estimates, and observed these transformations to account for about 10% of the total variance.

Figure 5.2 shows convergence results when training the last layer of a 50-layer Residual network (He et al., 2016) that has been pre-trained on ImageNet. Here, we consider the common scenario of leveraging a deep model trained on a large dataset as a feature extractor in order to learn a new classifier on a different small dataset, where it would be difficult to train such a model from scratch. To simulate this setting, we consider a binary classification task on a small dataset of 100 images of size 256x256 taken from the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012, which we crop to 224x224 before performing random adjustments to brightness, saturation, hue and contrast. As in the STL-10 experiments, the gains of S-MISO over other methods are of about one order of magnitude in suboptimality, as predicted by Table 5.2.

Dropout on gene expression data. We trained a binary logistic regression model on the breast cancer dataset of van de Vijver et al. (2002), with different Dropout rates δ, *i.e.*, where at every iteration, each coordinate ξ_j of a feature vector ξ is set to zero independently with probability δ and to $\xi_j/(1-\delta)$ otherwise. The dataset consists of 295 vectors of dimension 8 141 of gene expression data, which we normalize in ℓ_2 norm. Figure 5.3 (top) compares S-MISO with SGD and N-SAGA for three values of δ, as a way to control the variance of the perturbations. We include a Dropout rate of 0.01 to illustrate the impact of δ on the algorithms and study the influence of the perturbation variance σ_p^2, even though this value of δ is less relevant for the task. The plots show very clearly how the variance induced by the perturbations affects the convergence of S-MISO, giving suboptimality values that may be orders of magnitude smaller than SGD. This behavior is consistent with the theoretical convergence rate established in Section 5.3 and shows that the practice matches the theory.

Dropout on movie review sentiment analysis data. We trained a binary classifier with a squared hinge loss on the IMDB dataset (Maas et al., 2011) with different Dropout rates δ. We use the labeled part of the IMDB dataset, which consists of 25K training and 250K testing movie reviews, represented as 89 527-dimensional sparse bag-of-words vectors. In contrast to the previous experiments, we do not normalize the representations, which have great variability in their norms, in particular, the maximum Lipschitz constant across training points is roughly 100 times larger than the average one. Figure 5.3 (bottom) compares non-uniform sampling versions of S-MISO (see Ap-

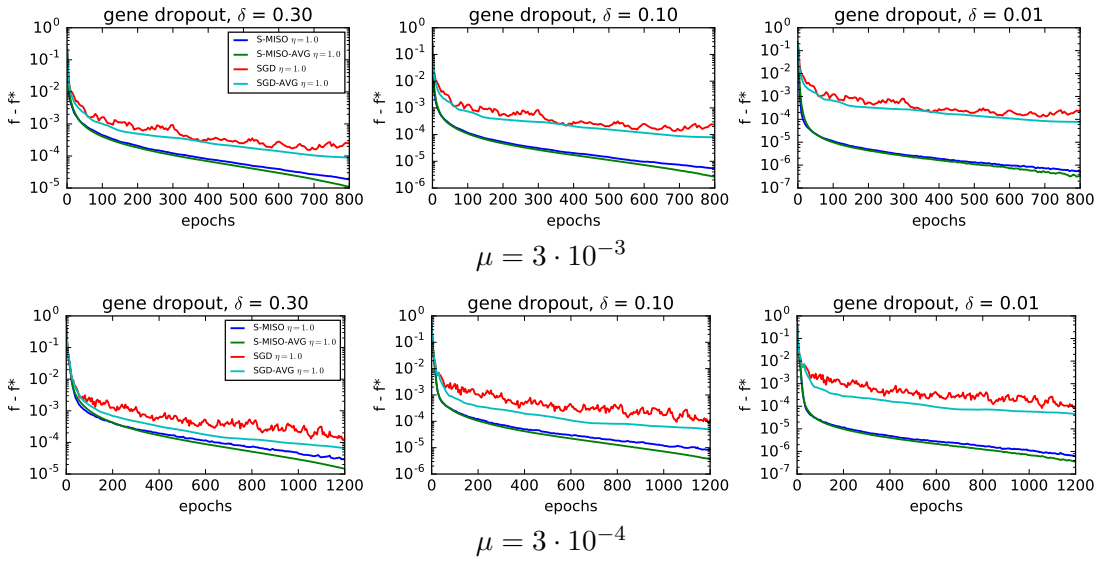


Figure 5.4: Comparison of S-MISO and SGD with averaging, for different condition numbers (controlled by μ) and different Dropout rates δ. We begin step-size decay and averaging at epoch 3 (top) and 30 (bottom).

pendix 5.4) and SGD (see Appendix 5.C) with their uniform sampling counterparts as well as N-SAGA. Note that we use a large step-size $\eta = 10$ for the uniform sampling algorithms, since $\eta = 1$ was significantly slower for all methods, likely due to outliers in the dataset. In contrast, the non-uniform sampling algorithms required no tuning and just use $\eta = 1$. The curves clearly show that S-MISO-NU has a much faster convergence in the initial phase, thanks to the larger step-size allowed by non-uniform sampling, and later converges similarly to S-MISO, *i.e.*, at a much faster rate than SGD when the perturbations are small. The value of μ used in the experiments was chosen by cross-validation, and the use of Dropout gave improvements in test accuracy from 88.51% with no dropout to $88.68 \pm 0.03\%$ with $\delta = 0.1$ and $88.86 \pm 0.11\%$ with $\delta = 0.3$ (based on 10 different runs of S-MISO-NU after 400 epochs).

Effect of iterate averaging. Figure 5.4 shows a comparison of S-MISO and SGD with the averaging scheme proposed in Theorem 5.3 (see Appendix 5.C for comments on how it applies to SGD), on the breast cancer dataset presented in Section 5.5, for different values of the regularization μ (and thus of the condition number $\kappa = L/\mu$), and Dropout rates δ. We can see that the averaging scheme gives some small improvements for both methods, and that the improvements are more significant when the problem is more ill-conditioned (Figure 5.4, bottom). We note that the time at which we start averaging can have significant impact on the convergence, in particular, starting too early can significantly slow down the initial convergence, as commonly noticed for stochastic gradient methods (see, *e.g.*, Nemirovski et al., 2009).

Appendix

5.A Proofs for the Smooth Case (Section 5.3)

5.A.1 Proof of Proposition 5.1 (Recursion on Lyapunov function C_t)

We begin by stating the following lemma, which extends a key result of variance reduction methods (see, *e.g.*, Johnson and Zhang, 2013) to the situation considered in this chapter, where one only has access to noisy estimates of the gradients of each f_i.

Lemma 5.A.1. *Let i be uniformly distributed in $\{1, \dots, n\}$ and ρ according to a perturbation distribution Γ. Under assumption (A2) on the functions $\tilde{f}_1, \dots, \tilde{f}_n$ and their expectations $f_1, \dots, f_n$, we have, for all $x \in \mathbb{R}^p$,*

$$\mathbb{E}_{i,\rho}[\|\nabla \tilde{f}_i(x, \rho) - \nabla f_i(x^*)\|^2] \leq 4L(f(x) - f(x^*)) + 2\sigma_p^2.$$

Proof. We have

$$\begin{aligned}
&\|\nabla \tilde{f}_i(x, \rho) - \nabla f_i(x^*)\|^2 \\
&\leq 2\|\nabla \tilde{f}_i(x, \rho) - \nabla \tilde{f}_i(x^*, \rho)\|^2 + 2\|\nabla \tilde{f}_i(x^*, \rho) - \nabla f_i(x^*)\|^2 \\
&\leq 4L(\tilde{f}_i(x, \rho) - \tilde{f}_i(x^*, \rho) - \langle \nabla \tilde{f}_i(x^*, \rho), x - x^* \rangle) + 2\|\nabla \tilde{f}_i(x^*, \rho) - \nabla f_i(x^*)\|^2.
\end{aligned}$$

The first inequality comes from the simple relation $\|u+v\|^2 + \|u-v\|^2 = 2\|u\|^2 + 2\|v\|^2$. The second inequality follows from the smoothness of $\tilde{f}_i(\cdot, \rho)$, in particular we used the classical relation

$$g(y) \geq g(x) + \langle \nabla g(x), y - x \rangle + \frac{1}{2L}\|\nabla g(y) - \nabla g(x)\|^2,$$

which is known to hold for any convex and L-smooth function g (see, *e.g.*, Nesterov, 2004, Theorem 2.1.5). The result follows by taking expectations on i and ρ and noting that $\mathbb{E}_{i,\rho}[\nabla \tilde{f}_i(x^*, \rho)] = \nabla f(x^*) = 0$, as well as the definition of σ_p^2. □

We now proceed with the proof of Proposition 5.1.

Proof. Define the quantities

$$A_t = \frac{1}{n}\sum_{i=1}^{n} \|z_i^t - z_i^*\|^2$$
$$\text{and} \quad B_t = \frac{1}{2}\|x_t - x^*\|^2.$$

The proof successively describes recursions on A_t, B_t, and eventually C_t.

Recursion on A_t. We have

$$\begin{aligned} A_t - A_{t-1} &= \frac{1}{n}(\|z_{i_t}^t - z_{i_t}^*\|^2 - \|z_{i_t}^{t-1} - z_{i_t}^*\|^2) \\ &= \frac{1}{n}\left(\left\|(1-\alpha_t)(z_{i_t}^{t-1} - z_{i_t}^*) + \alpha_t\left(x_{t-1} - \frac{1}{\mu}\nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - z_{i_t}^*\right)\right\|^2 - \|z_{i_t}^{t-1} - z_{i_t}^*\|^2\right) \\ &= \frac{1}{n}\left(-\alpha_t\|z_{i_t}^{t-1} - z_{i_t}^*\|^2 + \alpha_t\left\|x_{t-1} - \frac{1}{\mu}\nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - z_{i_t}^*\right\|^2 - \alpha_t(1-\alpha_t)\|v_t\|^2\right), \end{aligned} \tag{5.22}$$

where we first use the definition of z_i^t in (5.3), then the relation $\|(1-\lambda)u + \lambda v\|^2 = (1-\lambda)\|u\|^2 + \lambda\|v\|^2 - \lambda(1-\lambda)\|u-v\|^2$, and the definition of v_t given in (5.7). A similar relation is derived in the proof of SDCA without duality (Shalev-Shwartz, 2016). Using the definition of z_i^*, the second term can be expanded as

$$\begin{aligned} \left\|x_{t-1} - \frac{1}{\mu}\nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - z_{i_t}^*\right\|^2 &= \left\|x_{t-1} - x^* - \frac{1}{\mu}(\nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - \nabla f_{i_t}(x^*))\right\|^2 \\ &= \|x_{t-1} - x^*\|^2 - \frac{2}{\mu}\langle x_{t-1} - x^*, \nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - \nabla f_{i_t}(x^*)\rangle \\ &\quad + \frac{1}{\mu^2}\left\|\nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - \nabla f_{i_t}(x^*))\right\|^2. \end{aligned} \tag{5.23}$$

We then take conditional expectations with respect to $\mathcal{F}_{t-1}$, defined in Section 5.2. Unless otherwise specified, we will simply write $\mathbb{E}[\cdot]$ instead of $\mathbb{E}[\cdot|\mathcal{F}_{t-1}]$ for these conditional expectations in the rest of the proof.

$$\begin{aligned} \mathbb{E}\left[\left\|x_{t-1} - \frac{1}{\mu}\nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - z_{i_t}^*\right\|^2\right] &\le \|x_{t-1} - x^*\|^2 - \frac{2}{\mu}\langle x_{t-1} - x^*, \nabla f(x_{t-1})\rangle \\ &\quad + \frac{4L}{\mu^2}(f(x_{t-1}) - f(x^*)) + \frac{2\sigma_p^2}{\mu^2} \\ &\le \|x_{t-1} - x^*\|^2 - \frac{2}{\mu}(f(x_{t-1}) - f(x^*) + \frac{\mu}{2}\|x_{t-1} - x^*\|^2) \\ &\quad + \frac{4L}{\mu^2}(f(x_{t-1}) - f(x^*)) + \frac{2\sigma_p^2}{\mu^2} \\ &= \frac{2(2\kappa-1)}{\mu}(f(x_{t-1}) - f(x^*)) + \frac{2\sigma_p^2}{\mu^2}, \end{aligned}$$

where we used $\mathbb{E}[\nabla f_{i_t}(x^*)] = \nabla f(x^*) = 0$, Lemma 5.A.1, and the μ-strong convexity of f. Taking expectations on the previous relation on A_t yields

$$\begin{aligned} \mathbb{E}[A_t - A_{t-1}] &= -\frac{\alpha_t}{n}A_{t-1} + \frac{\alpha_t}{n}\,\mathbb{E}\left[\left\|x_{t-1} - \frac{1}{\mu}\nabla\tilde{f}_{i_t}(x_{t-1},\rho_t) - z_{i_t}^*\right\|^2\right] - \frac{\alpha_t(1-\alpha_t)}{n}\,\mathbb{E}[\|v_t\|^2] \\ &\le -\frac{\alpha_t}{n}A_{t-1} + \frac{2\alpha_t(2\kappa-1)}{n\mu}(f(x_{t-1}) - f(x^*)) - \frac{\alpha_t(1-\alpha_t)}{n}\,\mathbb{E}[\|v_t\|^2] + \frac{2\alpha_t\sigma_p^2}{n\mu^2}. \end{aligned} \tag{5.24}$$

Recursion on B_t. Separately, we have

$$\begin{aligned}\|x_t - x^*\|^2 &= \left\|x_{t-1} - x^* + \frac{\alpha_t}{n} v_t\right\|^2 \\ &= \|x_{t-1} - x^*\|^2 + \frac{2\alpha_t}{n}\langle x_{t-1} - x^*, v_t\rangle + \left(\frac{\alpha_t}{n}\right)^2 \|v_t\|^2 \\ \mathbb{E}[\|x_t - x^*\|^2] &= \|x_{t-1} - x^*\|^2 - \frac{2\alpha_t}{n\mu}\langle x_{t-1} - x^*, \nabla f(x_{t-1})\rangle + \left(\frac{\alpha_t}{n}\right)^2 \mathbb{E}[\|v_t\|^2] \\ &\leq \|x_{t-1} - x^*\|^2 - \frac{2\alpha_t}{n\mu}(f(x_{t-1}) - f(x^*) + \frac{\mu}{2}\|x_{t-1} - x^*\|^2) + \left(\frac{\alpha_t}{n}\right)^2 \mathbb{E}[\|v_t\|^2],\end{aligned}$$

using that $\mathbb{E}[v_t] = -\frac{1}{\mu}\nabla f(x_{t-1})$ and the strong convexity of f. This gives

$$\mathbb{E}[B_t - B_{t-1}] \leq -\frac{\alpha_t}{n} B_{t-1} - \frac{\alpha_t}{n\mu}(f(x_{t-1}) - f(x^*)) + \frac{1}{2}\left(\frac{\alpha_t}{n}\right)^2 \mathbb{E}[\|v_t\|^2]. \tag{5.25}$$

Recursion on C_t. If we consider $C_t = p_t A_t + B_t$ and $C'_{t-1} = p_t A_{t-1} + B_{t-1}$, combining (5.24) and (5.25) yields

$$\begin{aligned}&\mathbb{E}[C_t - C'_{t-1}] \leq \\ &-\frac{\alpha_t}{n} C'_{t-1} + \frac{2\alpha_t}{n\mu}(p_t(2\kappa - 1) - \frac{1}{2})(f(x_{t-1}) - f(x^*)) + \frac{\alpha_t}{n}\left(\frac{\alpha_t}{2n} - p_t(1 - \alpha_t)\right)\mathbb{E}[\|v_t\|^2] + \frac{2\alpha_t p_t \sigma_p^2}{n\mu^2}.\end{aligned}$$

If we take $p_t = \frac{\alpha_t}{n}$, and if $(\alpha_t)_{t\geq 1}$ is a decreasing sequence satisfying (5.9), then the factors in front of $f(x_{t-1}) - f(x^*)$ and $\mathbb{E}[\|v_t\|^2]$ are non-positive and we get

$$\mathbb{E}[C_t] \leq \left(1 - \frac{\alpha_t}{n}\right) C'_{t-1} + 2\left(\frac{\alpha_t}{n}\right)^2 \frac{\sigma_p^2}{\mu^2}.$$

Finally, since $\alpha_t \leq \alpha_{t-1}$, we have $C'_{t-1} \leq C_{t-1}$. After taking total expectations on $\mathcal{F}_{t-1}$, we are left with the desired recursion.

□

5.A.2 Proof of Theorem 5.2 (Convergence of C_t under decreasing step-sizes)

We prove the theorem with more general step-sizes:

Theorem 5.A.1 (Convergence of Lyapunov function)**.** *Let the sequence of step-sizes $(\alpha_t)_{t\geq 1}$ be defined by $\alpha_t = \frac{\beta n}{\gamma + t}$ with $\beta > 1$ and $\gamma \geq 0$ such that α_1 satisfies* (5.9)*. For all $t \geq 0$, it holds that*

$$\mathbb{E}[C_t] \leq \frac{\nu}{\gamma + t + 1} \qquad \textit{where} \qquad \nu := \max\left\{\frac{2\beta^2\sigma_p^2}{\mu^2(\beta - 1)}, (\gamma + 1)C_0\right\}. \tag{5.26}$$

In particular, taking $\beta = 2$ as in Theorem 5.2 can only improve the constant ν in the convergence rate.

Proof. Let us proceed by induction. We have $C_0 \leq \nu/(\gamma+1)$ by definition of ν. For $t \geq 1$,

$$\begin{aligned}
\mathbb{E}[C_t] &\leq \left(1 - \frac{\alpha_t}{n}\right) \mathbb{E}[C_{t-1}] + 2\left(\frac{\alpha_t}{n}\right)^2 \frac{\sigma_p^2}{\mu^2} \\
&\leq \left(1 - \frac{\beta}{\hat{t}}\right) \frac{\nu}{\hat{t}} + \frac{2\beta^2\sigma_p^2}{\hat{t}^2\mu^2} \quad \text{(with } \hat{t} := \gamma + t\text{)} \\
&= \left(\frac{\hat{t}-\beta}{\hat{t}^2}\right)\nu + \frac{2\beta^2\sigma_p^2}{\hat{t}^2\mu^2} \\
&= \left(\frac{\hat{t}-1}{\hat{t}^2}\right)\nu - \left(\frac{\beta-1}{\hat{t}^2}\right)\nu + \frac{2\beta^2\sigma_p^2}{\hat{t}^2\mu^2} \\
&\leq \left(\frac{\hat{t}-1}{\hat{t}^2}\right)\nu \leq \frac{\nu}{\hat{t}+1},
\end{aligned}$$

where the last two inequalities follow from the definition of ν and from $\hat{t}^2 \geq (\hat{t}+1)(\hat{t}-1)$. □

5.A.3 Proof of Theorem 5.3 (Convergence in function values under iterate averaging)

Proof. From the proof of Proposition 5.1, we have

$$\mathbb{E}[C_t] \leq \left(1 - \frac{\alpha_t}{n}\right) \mathbb{E}[C_{t-1}] + \frac{2\alpha_t}{n\mu}\left(\frac{\alpha_t}{n}(2\kappa - 1) - \frac{1}{2}\right) \mathbb{E}[f(x_{t-1}) - f(x^*)] + 2\left(\frac{\alpha_t}{n}\right)^2 \frac{\sigma_p^2}{\mu^2}.$$

The result holds because the choice of step-sizes $(\alpha_t)_{t\geq 1}$ safisfies the assumptions of Proposition 5.1. With our new choice of step-sizes, we have the stronger bound

$$\frac{\alpha_t}{n}(2\kappa - 1) - \frac{1}{2} \leq -\frac{1}{4}.$$

After rearranging, we obtain

$$\frac{\alpha_t}{2n\mu} \mathbb{E}[f(x_{t-1}) - f(x^*)] \leq \left(1 - \frac{\alpha_t}{n}\right) \mathbb{E}[C_{t-1}] - \mathbb{E}[C_t] + 2\left(\frac{\alpha_t}{n}\right)^2 \frac{\sigma_p^2}{\mu^2}. \tag{5.27}$$

Dividing by $\frac{\alpha_t}{2n\mu}$ gives

$$\begin{aligned}
\mathbb{E}[f(x_{t-1}) - f(x^*)] &\leq 2\mu\left[\left(\frac{n}{\alpha_t} - 1\right)\mathbb{E}[C_{t-1}] - \frac{n}{\alpha_t}\mathbb{E}[C_t]\right] + 4\frac{\alpha_t}{n}\frac{\sigma_p^2}{\mu} \\
&= \mu\left((\gamma + t - 2)\,\mathbb{E}[C_{t-1}] - (\gamma + t)\,\mathbb{E}[C_t]\right) + \frac{8}{\gamma + t}\frac{\sigma_p^2}{\mu}.
\end{aligned}$$

Multiplying by $(\gamma + t - 1)$ yields

$$\begin{aligned}
&(\gamma + t - 1)\,\mathbb{E}[f(x_{t-1}) - f(x^*)] \\
&\quad\leq \mu\left((\gamma + t - 1)(\gamma + t - 2)\,\mathbb{E}[C_{t-1}] - (\gamma + t)(\gamma + t - 1)\,\mathbb{E}[C_t]\right) + \frac{8(\gamma + t - 1)}{\gamma + t}\frac{\sigma_p^2}{\mu} \\
&\quad\leq \mu\left((\gamma + t - 1)(\gamma + t - 2)\,\mathbb{E}[C_{t-1}] - (\gamma + t)(\gamma + t - 1)\,\mathbb{E}[C_t]\right) + \frac{8\sigma_p^2}{\mu}.
\end{aligned}$$

By summing the above inequality from $t = 1$ to $t = T$, we have a telescoping sum that simplifies as follows:

$$\mathbb{E}\left[\sum_{t=1}^{T}(\gamma+t-1)(f(x_{t-1})-f(x^*))\right] \leq \mu\left(\gamma(\gamma-1)C_0-(\gamma+T)(\gamma+T-1)\,\mathbb{E}[C_T]\right)+\frac{8T\sigma_p^2}{\mu}$$
$$\leq \mu\gamma(\gamma-1)C_0+\frac{8T\sigma_p^2}{\mu}.$$

Dividing by $\sum_{t=1}^{T}(\gamma+t-1) = (2T\gamma+T(T-1))/2$ and using Jensen's inequality on $f(\bar{x}_T)$ gives the desired result. □

5.B Proofs for Composite Objectives and Non-Uniform Sampling (Section 5.4)

We recall here the updates to the lower bounds d_i^t in the setting of this section, which are analogous to (5.6) but with non-uniform weights and stochastic perturbations,: for $i = i_t$, we have

$$d_i^t(x) = \Big(1-\frac{\alpha_t}{q_i n}\Big)d_i^{t-1}(x)+\frac{\alpha_t}{q_i n}\Big(\tilde{f}_i(x_{t-1},\rho_t)+\langle\nabla\tilde{f}_i(x_{t-1},\rho_t),x-x_{t-1}\rangle+\frac{\mu}{2}\|x-x_{t-1}\|^2\Big), \tag{5.28}$$

and $d_i^t(x) = d_i^{t-1}(x)$ otherwise.

5.B.1 Proof of Lemma 5.4 (Bound on the iterates)

Proof. Let $F_t(x) := \frac{1}{n}\sum_{i=1}^{n} f_i^t(x) + h(x)$, where $f_i^0(x) = \tilde{f}_i(x,\tilde{\rho}_i)$ (where $\tilde{\rho}_i$ is used in (5.16)), and f_i^t is updated analogously to d_i^t as follows:

$$f_i^t(x) = \begin{cases}(1-\frac{\alpha_t}{q_i n})f_i^{t-1}(x)+\frac{\alpha_t}{q_i n}\tilde{f}_i(x,\rho_t), & \text{if } i = i_t\\ f_i^{t-1}(x), & \text{otherwise.}\end{cases}$$

By induction, we have

$$F_t(x^*) \geq D_t(x^*) \geq D_t(x_t)+\frac{\mu}{2}\|x_t-x^*\|^2, \tag{5.29}$$

where the last inequality follows from the μ-strong convexity of D_t and the fact that x_t is its minimizer.

Again by induction, we now show that $\mathbb{E}[F_t(x^*)] = F(x^*)$. Indeed, we have $\mathbb{E}[F_0(x^*)] = F(x^*)$ by construction, then

$$F_t(x^*) = F_{t-1}(x^*)+\frac{\alpha_t}{q_i n^2}(\tilde{f}_{i_t}(x^*,\rho_t)-f_{i_t}^{t-1}(x^*))$$
$$\mathbb{E}[F_t(x^*)|\mathcal{F}_{t-1}] = F_{t-1}(x^*)+\frac{\alpha_t}{n}(f(x^*)-\frac{1}{n}\sum_{i=1}^{n}f_i^{t-1}(x^*))$$
$$= F_{t-1}(x^*)+\frac{\alpha_t}{n}(F(x^*)-F_{t-1}(x^*)),$$

After taking total expectations and using the induction hypothesis, we obtain $\mathbb{E}[F_t(x^*)] = F(x^*)$, and the result follows from (5.29). □

5.B.2 Proof of Proposition 5.5 (Recursion on Lyapunov function C_t^q)

We begin by presenting a lemma that plays a similar role to Lemma 5.A.1 in our proof, but considers the composite objective and takes into account the new strong convexity and non-uniformity assumptions.

Lemma 5.B.1. *Let $i \sim q$, where q is the sampling distribution, and ρ be a random perturbation. Under assumptions (A4-5) on the functions $\tilde{f}_1, \dots, \tilde{f}_n$ and their expectations $f_1, \dots, f_n$, we have, for all $x \in \mathbb{R}^p$,*

$$\mathbb{E}_{i,\rho}\left[\frac{1}{(q_i n)^2}\|\nabla \tilde{f}_i(x,\rho) - \mu x - (\nabla f_i(x^*) - \mu x^*)\|^2\right] \leq 4L_q(F(x) - F(x^*)) + 2\sigma_q^2,$$

with $L_q = \max_i \frac{L_i - \mu}{q_i n}$ and $\sigma_q^2 = \frac{1}{n}\sum_i \frac{\sigma_i^2}{q_i n}$.

Proof. Since $\tilde{f}_i(\cdot,\rho)$ is μ-strongly convex and L_i-smooth, we have that $\tilde{f}_i(\cdot,\rho) - \frac{\mu}{2}\|\cdot\|^2$ is convex and $(L_i - \mu)$-smooth (this is a straightforward consequence of Nesterov, 2004, Eq. 2.1.9 and 2.1.22). Then, by denoting by F_i the quantity $2\|\nabla \tilde{f}_i(x^*,\rho) - \nabla f_i(x^*)\|^2$, we have

$$\begin{aligned}
&\|\nabla \tilde{f}_i(x,\rho) - \mu x - (\nabla f_i(x^*) - \mu x^*)\|^2 \\
&\quad\leq 2\|\nabla \tilde{f}_i(x,\rho) - \mu x - (\nabla \tilde{f}_i(x^*,\rho) - \mu x^*)\|^2 + 2\|\nabla \tilde{f}_i(x^*,\rho) - \nabla f_i(x^*)\|^2 \\
&\quad\leq 4(L_i - \mu)\left(\tilde{f}_i(x,\rho) - \frac{\mu}{2}\|x\|^2 - \tilde{f}_i(x^*,\rho) + \frac{\mu}{2}\|x^*\|^2 - \langle \nabla \tilde{f}_i(x^*,\rho) - \mu x^*, x - x^*\rangle\right) + F_i \\
&\quad= 4(L_i - \mu)\left(\tilde{f}_i(x,\rho) - \tilde{f}_i(x^*,\rho) - \langle \nabla \tilde{f}_i(x^*,\rho), x - x^*\rangle - \frac{\mu}{2}\|x - x^*\|^2\right) + F_i \\
&\quad\leq 4(L_i - \mu)\left(\tilde{f}_i(x,\rho) - \tilde{f}_i(x^*,\rho) - \langle \nabla \tilde{f}_i(x^*,\rho), x - x^*\rangle\right) + F_i.
\end{aligned}$$

The first inequality comes from the classical relation $\|u+v\|^2 + \|u-v\|^2 = 2\|u\|^2 + 2\|v\|^2$. The second inequality follows from the convexity and $(L_i - \mu)$-smoothness of $\tilde{f}_i(\cdot,\rho) - \frac{\mu}{2}\|\cdot\|^2$. Dividing by $(q_i n)^2$ and taking expectations yields

$$\begin{aligned}
&\mathbb{E}_{i,\rho}\left[\frac{1}{(q_i n)^2}\ \|\nabla \tilde{f}_i(x,\rho) - \mu x - (\nabla f_i(x^*) - \mu x^*)\|^2\right] \\
&\qquad\leq 4\sum_{i=1}^n \frac{q_i(L_i - \mu)}{(q_i n)^2}\left(f_i(x) - f_i(x^*) - \langle \nabla f_i(x^*), x - x^*\rangle\right) + 2\sum_{i=1}^n \frac{q_i}{(q_i n)^2}\sigma_i^2 \\
&\qquad= 4\frac{1}{n}\sum_{i=1}^n \frac{L_i - \mu}{q_i n}\left(f_i(x) - f_i(x^*) - \langle \nabla f_i(x^*), x - x^*\rangle\right) + 2\frac{1}{n}\sum_{i=1}^n \frac{\sigma_i^2}{q_i n} \\
&\qquad\leq 4L_q(f(x) - f(x^*) - \langle \nabla f(x^*), x - x^*\rangle) + 2\sigma_q^2 \\
&\qquad\leq 4L_q(f(x) - f(x^*) + h(x) - h(x^*)) + 2\sigma_q^2 = 4L_q(F(x) - F(x^*)) + 2\sigma_q^2,
\end{aligned}$$

where the last inequality follows from the optimality of x^*, which implies that $-\nabla f(x^*) \in \partial h(x^*)$, and in turn implies $\langle -\nabla f(x^*), x - x^*\rangle \leq h(x) - h(x^*)$ by convexity of h. □

We can now proceed with the proof of Proposition 5.5.

Proof. Define the quantities

$$A_t = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{q_i n}\|z_i^t - z_i^*\|^2$$
$$\text{and} \quad B_t = F(x^*) - D_t(x_t).$$

The proof successively describes recursions on A_t, B_t, and eventually C_t (we drop the superscript in C_t^q for simplicity), using the same approach as for the proof of Proposition 5.1.

Recursion on A_t. Using similar techniques as in the proof of Proposition 5.1, we have

$$\begin{aligned}
&A_t - A_{t-1} \\
&= \frac{1}{n}\left(\frac{1}{q_{i_t}n}\|z_{i_t}^t - z_{i_t}^*\|^2 - \frac{1}{q_{i_t}n}\|z_{i_t}^{t-1} - z_{i_t}^*\|^2\right) \\
&= \frac{1}{n}\left(\frac{1}{q_{i_t}n}\left\|\left(1-\frac{\alpha_t}{q_{i_t}n}\right)(z_{i_t}^{t-1} - z_{i_t}^*) + \frac{\alpha_t}{q_{i_t}n}\left(x_{t-1} - \frac{1}{\mu}\nabla \tilde{f}_{i_t}(x_{t-1},\rho_t) - z_{i_t}^*\right)\right\|^2 - \frac{1}{q_{i_t}n}\|z_{i_t}^{t-1} - z_{i_t}^*\|^2\right) \\
&= \frac{1}{n}\Big(-\frac{\alpha_t}{(q_{i_t}n)^2}\|z_{i_t}^{t-1} - z_{i_t}^*\|^2 + \frac{\alpha_t}{(q_{i_t}n)^2}\left\|x_{t-1} - \frac{1}{\mu}\nabla \tilde{f}_{i_t}(x_{t-1},\rho_t) - z_{i_t}^*\right\|^2 - \frac{\alpha_t}{(q_{i_t}n)^2}\Big(1-\frac{\alpha_t}{q_{i_t}n}\Big)\|v_{i_t}^t\|^2\Big),
\end{aligned}$$

where $v_i^t := x_{t-1} - \frac{1}{\mu}\nabla \tilde{f}_i(x_{t-1},\rho_t) - z_i^{t-1}$. Taking conditional expectations w.r.t. $\mathcal{F}_{t-1}$ and using Lemma 5.B.1 to bound the second term yields

$$\begin{aligned}
&\mathbb{E}[A_t - A_{t-1}] \leq \\
&\quad -\frac{\alpha_t}{n}A_{t-1} + \frac{4\alpha_t L_q}{n\mu^2}(F(x_{t-1}) - F(x^*)) + \frac{2\alpha_t\sigma_q^2}{n\mu^2} - \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\alpha_t}{n}\frac{1}{q_i n}\left(1-\frac{\alpha_t}{q_i n}\right)\|v_i^t\|^2\right)
\end{aligned} \tag{5.30}$$

Recursion on B_t. We start by using a lemma from the proof of MISO-Prox (Lin et al., 2015, Lemma D.4), which only relies on the form of D_t and the fact that x_t minimizes it, and thus holds in our setting:

$$\begin{aligned}
D_t(x_t) &\geq D_t(x_{t-1}) - \frac{\mu}{2}\|\bar{z}_t - \bar{z}_{t-1}\|^2 \\
&= D_t(x_{t-1}) - \frac{\mu}{2(q_{i_t}n)^2}\left(\frac{\alpha_t}{n}\right)^2\|v_{i_t}^t\|^2
\end{aligned} \tag{5.31}$$

We then expand $D_t(x_{t-1})$ using (5.28) as follows:

$$\begin{aligned}
D_t(x_{t-1}) &= D_{t-1}(x_{t-1}) + \frac{\alpha_t}{n}\frac{1}{q_{i_t}n}\left(\tilde{f}_{i_t}(x_{t-1},\rho_t) - d_{i_t}^{t-1}(x_{t-1})\right) \\
&= D_{t-1}(x_{t-1}) + \frac{\alpha_t}{n}\frac{1}{q_{i_t}n}\left(\tilde{f}_{i_t}(x_{t-1},\rho_t) + h(x_{t-1}) - d_{i_t}^{t-1}(x_{t-1}) - h(x_{t-1})\right).
\end{aligned}$$

After taking conditional expections w.r.t. $\mathcal{F}_{t-1}$, (5.31) becomes

$$\mathbb{E}[D_t(x_t)] \geq D_{t-1}(x_{t-1}) + \frac{\alpha_t}{n}(F(x_{t-1}) - D_{t-1}(x_{t-1})) - \frac{\mu}{2n}\sum_{i=1}^{n}\left(\frac{\alpha_t}{n}\right)^2\frac{1}{q_i n}\|v_i^t\|^2.$$

Subtracting $F(x^*)$ and rearranging yields

$$\mathbb{E}[B_t - B_{t-1}] \leq -\frac{\alpha_t}{n} B_{t-1} - \frac{\alpha_t}{n}(F(x_{t-1}) - F(x^*)) + \frac{\mu}{2n} \sum_{i=1}^{n} \left(\frac{\alpha_t}{n}\right)^2 \frac{1}{q_i n} \|v_i^t\|^2. \quad (5.32)$$

Recursion on C_t. If we consider $C_t = \mu p_t A_t + B_t$ and $C'_{t-1} = \mu p_t A_{t-1} + B_{t-1}$, combining (5.30) and (5.32) yields

$$\mathbb{E}[C_t - C'_{t-1}] \leq -\frac{\alpha_t}{n} C'_{t-1} + \frac{2\alpha_t}{n}\Big(\frac{2p_t L_q}{\mu} - \frac{1}{2}\Big)(F(x_{t-1}) - F(x^*)) + \frac{\mu\alpha_t}{n^2} \sum_{i=1}^{n} \frac{\delta_i^t}{q_i n} \|v_i^t\|^2 + \frac{2\alpha_t p_t \sigma_q^2}{n\mu}, \quad (5.33)$$

with

$$\delta_i^t = \frac{\alpha_t}{2n} - p_t \left(1 - \frac{\alpha_t}{q_i n}\right).$$

If we take $p_t = \frac{\alpha_t}{n}$, and if $(\alpha_t)_{t \geq 1}$ is a decreasing sequence satisfying (5.19), then we obtain the desired recursion after noticing that $C'_{t-1} \leq C_{t-1}$ and taking total expectations on $\mathcal{F}_{t-1}$.

□

Note that if we take

$$\alpha_1 \leq \min\left\{\frac{n q_{\min}}{2}, \frac{n\mu}{8L_q}\right\},$$

then (5.33) yields

$$\mathbb{E}\left[\frac{C_t^q}{\mu}\right] \leq \left(1 - \frac{\alpha_t}{n}\right) \mathbb{E}\left[\frac{C_{t-1}^q}{\mu}\right] - \frac{\alpha_t}{2n\mu}(F(x_{t-1}) - F(x^*)) + 2\left(\frac{\alpha_t}{n}\right)^2 \frac{\sigma_q^2}{\mu^2}.$$

This relation takes the same form as Eq. (5.27), hence it is straightforward to adapt the proof of Theorem 5.3 to this setting, and the same iterate averaging scheme applies.

5.C Complexity Analysis of SGD

In this section, we provide a proof of a simple result for SGD in the smooth case, giving a recursion that depends on a variance condition at the optimum (in contrast to Bottou et al., 2018; Nemirovski et al., 2009, where this condition needs to hold everywhere), for a more natural comparison with S-MISO.

Proposition 5.C.1 (Simple SGD recursion with variance at optimum)**.** *Under assumptions (A1) and (A2), if $\eta_t \leq 1/2L$, then the SGD recursion $x_t := x_{t-1} - \eta_t \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t)$ satisfies*

$$B_t \leq (1 - \mu\eta_t) B_{t-1} + \eta_t^2 \sigma_{tot}^2,$$

where $B_t := \frac{1}{2}\mathbb{E}[\|x_t - x^*\|^2]$ and σ_{tot} is such that

$$\mathbb{E}_{i,\rho}\left[\|\nabla \tilde{f}_i(x^*, \rho)\|^2\right] \leq \sigma_{tot}^2.$$

Proof. We have

$$\begin{aligned}
\|x_t - x^*\|^2 &= \|x_{t-1} - x^*\|^2 - 2\eta_t \langle \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t), x_{t-1} - x^* \rangle + \eta_t^2 \|\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t)\|^2 \\
&\leq \|x_{t-1} - x^*\|^2 - 2\eta_t \langle \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t), x_{t-1} - x^* \rangle \\
&\quad + 2\eta_t^2 \|\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - \nabla \tilde{f}_{i_t}(x^*, \rho_t)\|^2 + 2\eta_t^2 \|\nabla \tilde{f}_{i_t}(x^*, \rho_t)\|^2 \\
\mathbb{E}\left[\|x_t - x^*\|^2\right] &\leq \|x_{t-1} - x^*\|^2 - 2\eta_t \langle \nabla f(x_{t-1}), x_{t-1} - x^* \rangle \\
&\quad + 2\eta_t^2 \, \mathbb{E}_{i_t, \rho_t}\left[\|\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - \nabla \tilde{f}_{i_t}(x^*, \rho_t)\|^2\right] + 2\eta_t^2 \, \mathbb{E}_{i_t, \rho_t}\left[\|\nabla \tilde{f}_{i_t}(x^*, \rho_t)\|^2\right] \\
(*) &\leq \|x_{t-1} - x^*\|^2 - 2\eta_t \left(f(x_{t-1}) - f(x^*) + \frac{\mu}{2}\|x_{t-1} - x^*\|^2\right) \\
&\quad + 4L\eta_t^2 \left(f(x_{t-1}) - f(x^*)\right) + 2\eta_t^2 \sigma_{\text{tot}}^2 \\
&= (1 - \mu\eta_t)\|x_{t-1} - x^*\|^2 - 2\eta_t(1 - 2L\eta_t)(f(x_{t-1}) - f(x^*)) + 2\eta_t^2 \sigma_{\text{tot}}^2,
\end{aligned}$$

where the expectation is taken with respect to the filtration $\mathcal{F}_{t-1}$ and the inequality $(*)$ follows from the strong convexity of f and $\mathbb{E}_{i,\rho}[\|\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - \nabla \tilde{f}_{i_t}(x^*, \rho_t)\|^2]$ is bounded by $2L(f(x_{t-1}) - f(x^*))$ as in the proof of Lemma 5.A.1. When $\eta_t \leq 1/2L$, the second term is non-positive and we obtain the desired result after taking total expectations. □

Note that when $\eta_t \leq 1/4L$, we have

$$\mathbb{E}\left[\|x_t - x^*\|^2\right] \leq (1 - \mu\eta_t)\, \mathbb{E}\left[\|x_{t-1} - x^*\|^2\right] - \eta_t(f(x_{t-1}) - f(x^*)) + 2\eta_t^2 \sigma_{\text{tot}}^2.$$

This takes a similar form to Eq. (5.27), and one can use the same iterate averaging scheme as Theorem 5.3 with step-sizes $\eta_t = 2/\mu(\gamma + t)$ by adapting the proof.

We now give a similar recursion for the proximal SGD algorithm (see, *e.g.*, Duchi and Singer, 2009). This allows us to apply the results of Theorem 5.2 and the step-size strategy mentioned in Section 5.3.

Proposition 5.C.2 (Simple recursion for proximal SGD with variance at optimum)**.** *Under assumptions (A1) and (A2), if $\eta_t \leq 1/2L$, then the proximal SGD recursion*

$$x_t := \operatorname{prox}_{\eta_t h}(x_{t-1} - \eta_t \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t))$$

satisfies

$$B_t \leq (1 - \mu\eta_t) B_{t-1} + \eta_t^2 \sigma_{tot}^2,$$

where $B_t := \frac{1}{2}\mathbb{E}[\|x_t - x^*\|^2]$ and σ_{tot} is such that

$$\mathbb{E}_{i,\rho}\left[\|\nabla \tilde{f}_i(x^*, \rho) - \nabla f(x^*)\|^2\right] \leq \sigma_{tot}^2.$$

Proof. We have

$$\begin{aligned}
&\|x_t - x^*\|^2 \\
&= \|\operatorname{prox}_{\eta_t h}(x_{t-1} - \eta_t \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t)) - \operatorname{prox}_{\eta_t h}(x^* - \eta_t \nabla f(x^*))\|^2 \\
&\leq \|x_{t-1} - \eta_t \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - x^* + \eta_t \nabla f(x^*)\|^2 \\
&= \|x_{t-1} - x^*\|^2 - 2\eta_t \langle \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - \nabla f(x^*), x_{t-1} - x^* \rangle + \eta_t^2 \|\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - \nabla f(x^*)\|^2 \\
&\leq \|x_{t-1} - x^*\|^2 - 2\eta_t \langle \nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - \nabla f(x^*), x_{t-1} - x^* \rangle \\
&\quad + 2\eta_t^2 \|\nabla \tilde{f}_{i_t}(x_{t-1}, \rho_t) - \nabla \tilde{f}_{i_t}(x^*, \rho_t)\|^2 + 2\eta_t^2 \|\nabla \tilde{f}_{i_t}(x^*, \rho_t) - \nabla f(x^*)\|^2,
\end{aligned}$$

where the first equality follows from the optimality of x^* and the following inequality follows from the non-expansiveness of proximal operators. Taking conditional expectations on $\mathcal{F}_{t-1}$ yields

$$\begin{aligned}
&\mathbb{E}\left[\|x_t - x^*\|^2 | \mathcal{F}_{t-1}\right] \\
&\leq \|x_{t-1} - x^*\|^2 - 2\eta_t \langle \nabla f(x_{t-1}) - \nabla f(x^*), x_{t-1} - x^* \rangle \\
&\quad + 2\eta_t^2 \, \mathbb{E}_{i_t,\rho_t}\left[\|\nabla \tilde{f}_{i_t}(x_{t-1},\rho_t) - \nabla \tilde{f}_{i_t}(x^*,\rho_t)\|^2\right] + 2\eta_t^2 \, \mathbb{E}_{i_t,\rho_t}\left[\|\nabla \tilde{f}_{i_t}(x^*,\rho_t) - \nabla f(x^*)\|^2\right] \\
(*) &\leq \|x_{t-1} - x^*\|^2 - 2\eta_t \left(f(x_{t-1}) - f(x^*) + \frac{\mu}{2}\|x_{t-1} - x^*\|^2 - \langle \nabla f(x^*), x_{t-1} - x^* \rangle \right) \\
&\quad + 4L\eta_t^2 \left(f(x_{t-1}) - f(x^*) - \langle \nabla f(x^*), x_{t-1} - x^* \rangle \right) + 2\eta_t^2 \sigma_{\text{tot}}^2 \\
&= (1 - \mu\eta_t)\|x_{t-1} - x^*\|^2 - 2\eta_t(1 - 2L\eta_t)(f(x_{t-1}) - f(x^*) - \langle \nabla f(x^*), x_{t-1} - x^* \rangle) + 2\eta_t^2 \sigma_{\text{tot}}^2,
\end{aligned}$$

where inequality $(*)$ follows from the μ-strong convexity of f and $\mathbb{E}_{i,\rho}[\|\nabla \tilde{f}_{i_t}(x_{t-1},\rho_t) - \nabla \tilde{f}_{i_t}(x^*,\rho_t)\|^2]$ is bounded by $2L(f(x_{t-1}) - f(x^*) - \langle \nabla f(x^*), x_{t-1} - x^* \rangle)$ as in the proof of Lemma 5.B.1. By convexity of f, we have $f(x_{t-1}) - f(x^*) - \langle \nabla f(x^*), x_{t-1} - x^* \rangle \geq 0$, hence the second term is non-positive when $\eta_t \leq 1/2L$. We conclude by taking total expectations. □

We note that Propositions 5.C.1 and 5.C.2 can be easily adapted to non-uniform sampling with sampling distribution q and step-sizes $\eta_t / q_{i_t} n$, leading to step-size conditions $\eta_t \leq 1/2L_q$, with $L_q = \max_i \frac{L_i}{q_i n}$ and variance $\sigma_{q,tot}^2 = \mathbb{E}_{i,\rho}[\frac{1}{(q_i n)^2}\|\nabla \tilde{f}_i(x^*,\rho) - \nabla f(x^*)\|^2]$.

Chapter 6

Conclusion and Perspectives

In this thesis, we presented various contributions on theoretical and practical foundations of deep learning, particularly convolutional networks, through the lens of kernel methods.

Our first contribution introduces of a functional space for studying convolutional architectures, by defining an appropriate multi-layer convolutional kernel on signals. This allows us to study theoretical guarantees on smoothness, invariance, stability to deformations, and signal preservation for the obtained signal representation. It also provides a way to measure complexity of certain CNNs with smooth activations, which are shown to be in the RKHS of the constructed kernels, through a study of their RKHS norm.

Our second contribution presents new practical regularization strategies for deep neural networks which are obtained through approximations of the RKHS norm for the multi-layer kernels introduced above. Our newly obtained methods lead to improved empirical generalization performance on small datasets from computer vision and computational biology, as well as state-of-the-art robustness to additive ℓ_2 perturbations on CIFAR10. Theoretical insights and guarantees are also provided for generative modeling and robust generalization, when the RKHS norm is appropriately controlled.

Our third contribution studies the inductive bias of gradient-based optimization in deep networks, focusing on a specific over-parameterized regime where a "lazy training" phenomenon occurs, with weights remaining very close to initialization. For regression problems, this regime leads to certain minimum-norm solutions in the RKHS of the so-called neural tangent kernel. We study the inductive bias in such a learning regime by analyzing smoothness, stability and approximation properties for this kernel, and comparing it to other deep kernels obtained, for instance, when only training the last layer of a deep network. We also observe that when the number of neurons is finite in this regime, smoothness and stability properties may only hold approximately for the neural network predictions, since it does not belong to the RKHS of the limiting (infinite-width) kernel. This may lead to some local instabilities, and might be a cause of adversarial examples.

Our fourth contribution considers stability and regularization more explicitly through data augmentation, and develops a stochastic optimization algorithm for the resulting objective function, in the strongly convex setting. This algorithm outperforms SGD by exploiting the finite-sum structure of the objective with variance reduction. More precisely, the convergence rate depends on the gradient variance induced by data aug-

mentation on a *single* example (which is unavoidable), which may be much smaller than the full gradient variance across all examples.

Perspectives. The contributions in this thesis raise several questions that would be interesting to tackle in future work, which we discuss below.

While Chapter 4 investigates approximation properties of the RKHS for kernels based on two-layer fully-connected networks following Bach (2017a), it would be interesting to obtain similar characterizations for multi-layer kernels, as well as for convolutional kernels. For the convolutional case, this raises the question of which functions can be approximated, and may require appropriate assumptions on the data distribution. While for the fully connected kernel there is a poor dependence on the dimensionality of the input data, what are appropriate assumptions on the target functions and on the data distribution which may avoid this curse in the convolutional case?

Other questions arise regarding deformation stability and approximation: we found that deep convolutional networks with small patches are near-translation invariant and stable to deformations; conversely, can all functions that are stable to deformations be approximated by a convolutional network? If so, can such networks be learned efficiently, perhaps using a kernel method with a convolutional kernel, or a form of optimization with over-parameterization? It has been observed that deep networks trained on standard image recognition datasets tend to focus on low-level details such as texture, rather than higher-level features such as shape; can such biases be controlled more explicitly through regularization, perhaps with suitable kernel design?

Finally, while kernels provide a useful tool for studying theoretical foundations of deep learning, including optimization of over-parameterized networks in a certain regime, it has been observed that such results might only provide a limited explanation for the success of deep networks. For instance, the lazy training regime where weights remain very close to initialization does not explain the common "feature selection" behavior of the first layers of CNNs which learn Gabor-like filters, as pointed out by Chizat et al. (2019). Recent papers have shown that optimization algorithms on certain over-parameterized models may achieve better sample complexity than kernel methods (*e.g.*, Allen-Zhu and Li, 2019; Wei et al., 2019). An interesting question for future work is to study generalization properties in a regime in which different layers may be optimized differently, some of which may be more "lazy" than others, as observed by Zhang et al. (2019), suggesting that a kernel-like behavior may still be partially present.

Appendix A

Software

Accompanying the contributions described in the chapters of this thesis, the following software packages were developed over the course of the PhD:

- *stochs* - `https://github.com/albietz/stochs`

  This package provides implementations of various stochastic optimization algorithms, including variance reduction methods and hybrid stochastic/finite sum methods including the *stochastic MISO* algorithm introduced in our contribution (Bietti and Mairal, 2017b). It is implemented in C++ using the Eigen library for linear algebra, and is mainly used in Python through a Cython interface.

- `https://github.com/albietz/kernel_reg`

  This package provides a PyTorch implementation of the regularizers introduced in (Bietti et al., 2019) for deep networks, and provides code for reproducing the results of the paper for learning on small datasets or with adversarial perturbations. The main project page above also provides updated results on robustness which use stronger attacks at test time, and show state-of-the-art performance on Cifar10.

- `https://github.com/albietz/ckn_kernel`

  This package implements exact computations for convolutional kernels such as those of Chapter 2 and the neural tangent kernels for convolutional networks described in Chapter 4. This was used for empirical studies of deformation stability. The main algorithm is implemented in C++ using dynamic programming, and can be accessed in Python through a Cython interface.

- For the empirical study of contextual bandits in (Bietti et al., 2018), several contributions were made to the open-source online learning library Vowpal Wabbit (VW),[1] specifically an implementation of the RegCB algorithm for contextual bandits, along with various improvements of other algorithms including Cover, Bagging, and ϵ-Greedy, as well as various contributions to the core parts of VW related to Contextual Bandits. The scripts used for evaluation are available at `https://github.com/albietz/cb_bakeoff`.

[1] See `https://github.com/VowpalWabbit/vowpal_wabbit` and `https://vowpalwabbit.org`.

Bibliography

M. Achab, A. Guilloux, S. Gaïffas, and E. Bacry. SGD with Variance Reduction beyond Empirical Risk Minimization. *arXiv preprint arXiv:1510.04822*, 2015.

B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831, 2015.

S. Allassonnière, Y. Amit, and A. Trouvé. Towards a coherent statistical framework for dense deformable template estimation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(1):3–29, 2007.

Z. Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *Journal of Machine Learning Research (JMLR)*, 18(1):8194–8244, 2017.

Z. Allen-Zhu and Y. Li. What can resnet learn efficiently, going beyond kernels? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Z. Allen-Zhu, Y. Yuan, and K. Sridharan. Exploiting the Structure: Stochastic Gradient Methods Using Raw Clusters. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Z. Allen-Zhu, Y. Li, and Y. Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019a.

Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019b.

S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, 25(17):3389–3402, 1997.

J. Andén and S. Mallat. Deep scattering spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, 2014.

F. Anselmi, L. Rosasco, C. Tan, and T. Poggio. Deep convolutional networks are hierarchical kernel machines. *preprint arXiv:1508.01084*, 2015.

F. Anselmi, L. Rosasco, and T. Poggio. On invariance and selectivity in representation learning. *Information and Inference*, 5(2):134–158, 2016.

M. Anthony and P. Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 2009.

M. Arbel, D. J. Sutherland, M. Bińkowski, and A. Gretton. On gradient regularizers for MMD GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.

S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

S. Arora, S. S. Du, W. Hu, Z. Li, R. Salakhutdinov, and R. Wang. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019a.

S. Arora, S. S. Du, W. Hu, Z. Li, and R. Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019b.

K. Atkinson and W. Han. *Spherical harmonics and approximations on the unit sphere: an introduction*, volume 2044. Springer Science & Business Media, 2012.

F. Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory (COLT)*, 2013.

F. Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research (JMLR)*, 18(19):1–53, 2017a.

F. Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research (JMLR)*, 18(21):1–38, 2017b.

F. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research (JMLR)*, 3(Jul):1–48, 2002.

F. Bach and M. I. Jordan. Predictive low-rank decomposition for kernel methods. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2005.

F. Bach and E. Moulines. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

F. Bach and E. Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate o (1/n). In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research (JMLR)*, 3(Nov):463–482, 2002.

P. L. Bartlett, O. Bousquet, S. Mendelson, et al. Local rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537, 2005.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

P. L. Bartlett, D. J. Foster, and M. Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

P. L. Bartlett, P. M. Long, G. Lugosi, and A. Tsigler. Benign overfitting in linear regression. *arXiv preprint arXiv:1906.11300*, 2019.

R. Basri, D. Jacobs, Y. Kasten, and S. Kritchman. The convergence rate of neural networks for learned functions of different frequencies. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

M. Belkin, D. Hsu, and P. Mitra. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018a.

M. Belkin, S. Ma, and S. Mandal. To understand deep learning we need to understand kernel learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018b.

M. Belkin, A. Rakhlin, and A. B. Tsybakov. Does data interpolation contradict statistical optimality? In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

Y. Bengio, N. L. Roux, P. Vincent, O. Delalleau, and P. Marcotte. Convex neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

A. Berlinet and C. Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer, 2004.

A. Bietti and J. Mairal. Invariance and stability of deep convolutional representations. In *Advances in Neural Information Processing Systems (NIPS)*, 2017a.

A. Bietti and J. Mairal. Stochastic optimization with variance reduction for infinite datasets with finite sum structure. In *Advances in Neural Information Processing Systems (NIPS)*, 2017b.

A. Bietti and J. Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *Journal of Machine Learning Research (JMLR)*, 20(25):1–49, 2019a.

A. Bietti and J. Mairal. On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019b.

A. Bietti, A. Agarwal, and J. Langford. A contextual bandit bake-off. *arXiv preprint arXiv:1802.04064*, 2018.

A. Bietti, G. Mialon, D. Chen, and J. Mairal. A kernel perspective for regularizing deep neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

B. Biggio and F. Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying MMD GANs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

L. Bo, K. Lai, X. Ren, and D. Fox. Object recognition with hierarchical kernel descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

L. Bottou and O. Bousquet. The Tradeoffs of Large Scale Learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: A survey of some recent advances. *ESAIM: probability and statistics*, 9:323–375, 2005.

J. Bouvrie, L. Rosasco, and T. Poggio. On invariance in hierarchical models. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4): 18–42, 2017.

J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(8):1872–1886, 2013.

J. Bruna, A. Szlam, and Y. LeCun. Learning stable group invariant representations with convolutional networks. *preprint arXiv:1301.3537*, 2013.

S. Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4), 2015.

Y. Cao and Q. Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

T. Ching et al. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface*, 15(141), 2018.

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

L. Chizat, E. Oyallon, and F. Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Y. Cho and L. K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2017.

A. Coates, H. Lee, and A. Y. Ng. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

J. Cohen, E. Rosenfeld, and Z. Kolter. Certified adversarial robustness via randomized smoothing. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

T. Cohen and M. Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning (ICML)*, 2016.

T. Cohen, M. Geiger, J. Koehler, and M. Welling. Spherical CNNs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American mathematical society*, 39(1):1–49, 2002.

A. Daniely. Sgd learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

A. Daniely, R. Frostig, and Y. Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

A. Daniely, R. Frostig, V. Gupta, and Y. Singer. Random features for compositional kernels. *preprint arXiv:1703.07872*, 2017.

A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems (NIPS)*, 2014a.

A. Defazio, J. Domke, and T. S. Caetano. Finito: A faster, permutable incremental gradient method for big data problems. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014b.

L. Devroye, L. Györfi, and G. Lugosi. *A probabilistic theory of pattern recognition*. Springer, 1996.

J. Diestel and J. J. Uhl. *Vector Measures*. American Mathematical Society, 1977.

A. Dieuleveut, F. Bach, et al. Nonparametric stochastic approximation with large step-sizes. *The Annals of Statistics*, 44(4):1363–1399, 2016.

A. Dieuleveut, N. Flammarion, and F. Bach. Harder, better, faster, stronger convergence rates for least-squares regression. *Journal of Machine Learning Research (JMLR)*, 18 (1):3520–3570, 2017.

H. Drucker and Y. Le Cun. Double backpropagation increasing generalization performance. In *International Joint Conference on Neural Networks (IJCNN)*, 1991.

S. S. Du, J. D. Lee, H. Li, L. Wang, and X. Zhai. Gradient descent finds global minima of deep neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019a.

S. S. Du, X. Zhai, B. Poczos, and A. Singh. Gradient descent provably optimizes over-parameterized neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019b.

J. C. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research (JMLR)*, 10:2899–2934, 2009.

J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Privacy aware learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

G. K. Dziugaite, D. M. Roy, and Z. Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2015.

C. Efthimiou and C. Frye. *Spherical harmonics in p dimensions*. World Scientific, 2014.

A. El Alaoui and M. Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry. Exploring the landscape of spatial robustness. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research (JMLR)*, 2:243–264, 2001.

S. Fischer and I. Steinwart. Sobolev norm learning rates for regularized least-squares algorithm. *arXiv preprint arXiv:1702.07254*, 2017.

G. B. Folland. *A course in abstract harmonic analysis*. Chapman and Hall/CRC, 2016.

A. Garriga-Alonso, L. Aitchison, and C. E. Rasmussen. Deep convolutional networks as shallow gaussian processes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

B. Ghorbani, S. Mei, T. Misiakiewicz, and A. Montanari. Linearized two-layers neural networks in high dimension. *arXiv preprint arXiv:1904.12191*, 2019.

N. Golowich, A. Rakhlin, and O. Shamir. Size-independent sample complexity of neural networks. In *Conference on Learning Theory (COLT)*, 2018.

A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research (JMLR)*, 13(Mar):723–773, 2012.

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of Wasserstein GANs. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

S. Gunasekar, B. E. Woodworth, S. Bhojanapalli, B. Neyshabur, and N. Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

S. Gunasekar, J. D. Lee, D. Soudry, and N. Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A distribution-free theory of nonparametric regression*. Springer, 2006.

B. Haasdonk and H. Burkhardt. Invariant kernel functions for pattern analysis and machine learning. *Machine learning*, 68(1):35–61, 2007.

T. Håndstad, A. J. Hestnes, and P. Sætrom. Motif kernel generated by genetic programming improves remote homology and fold detection. *BMC bioinformatics*, 8(1): 23, 2007.

T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2009.

T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2015.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms I: Fundamentals*. Springer science & business media, 1993.

T. Hofmann, A. Lucchi, S. Lacoste-Julien, and B. McWilliams. Variance Reduced Stochastic Gradient Descent with Neighbors. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

D. Hsu, S. Kakade, and T. Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 14(3), 2014.

A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

S. M. Kakade, K. Sridharan, and A. Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

J. Khim and P.-L. Loh. Adversarial risk bounds via function transformation. *arXiv preprint arXiv:1810.09519*, 2018.

G. Kimeldorf and G. Wahba. Some results on tchebycheffian spline functions. *Journal of mathematical analysis and applications*, 33(1):82–95, 1971.

V. Koltchinskii. Local rademacher complexities and oracle inequalities in risk minimization. *The Annals of Statistics*, 34(6):2593–2656, 2006.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30(1):1–50, 2002.

R. Kondor and S. Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

A. Kulunchakov and J. Mairal. Estimate sequences for stochastic composite optimization: Variance reduction, acceleration, and robustness to noise. *arXiv preprint arXiv:1901.08788*, 2019.

S. Lacoste-Julien, M. Schmidt, and F. Bach. A simpler approach to obtaining an O(1/t) convergence rate for the projected stochastic subgradient method. *arXiv preprint arXiv:1212.2002*, 2012.

G. Lan and Y. Zhou. An optimal randomized incremental gradient method. *Mathematical Programming*, 2017.

L. Landweber. An iteration formula for fredholm integral equations of the first kind. *American journal of mathematics*, 73(3):615–624, 1951.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as gaussian processes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

J. Lee, L. Xiao, S. S. Schoenholz, Y. Bahri, J. Sohl-Dickstein, and J. Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

Y. Li and Y. Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Y. Li, T. Ma, and H. Zhang. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. In *Conference on Learning Theory (COLT)*, 2018.

T. Liang and A. Rakhlin. Just interpolate: Kernel "ridgeless" regression can generalize. *Annals of Statistics*, 2019.

T. Liang, T. Poggio, A. Rakhlin, and J. Stokes. Fisher-Rao metric, geometry, and complexity of neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

H. Lin, J. Mairal, and Z. Harchaoui. A Universal Catalyst for First-Order Optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

J. Lin, A. Rudi, L. Rosasco, and V. Cevher. Optimal rates for spectral algorithms with least-squares regression over hilbert spaces. *Applied and Computational Harmonic Analysis*, 2018.

G. Loosli, S. Canu, and L. Bottou. Training invariant support vector machines using selective sampling. In *Large Scale Kernel Machines*, pages 301–320. MIT Press, Cambridge, MA., 2007.

C. Lyu, K. Huang, and H.-N. Liang. A unified gradient regularization family for adversarial examples. In *IEEE International Conference on Data Mining (ICDM)*, 2015.

A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 142–150. Association for Computational Linguistics, 2011.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

J. Mairal. Incremental Majorization-Minimization Optimization with Application to Large-Scale Machine Learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

J. Mairal. End-to-End Kernel Learning with Supervised Convolutional Kernel Networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid. Convolutional kernel networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

S. Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

E. Mammen and A. B. Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 27(6):1808–1829, 1999.

P. Massart and É. Nédélec. Risk bounds for statistical learning. *The Annals of Statistics*, 34(5):2326–2366, 2006.

A. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

S. Mei, A. Montanari, and P.-M. Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33): E7665–E7671, 2018.

S. Mei, T. Misiakiewicz, and A. Montanari. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. In *Conference on Learning Theory (COLT)*, 2019.

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018a.

T. Miyato, S.-i. Maeda, S. Ishii, and M. Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2018b.

G. Montavon, M. L. Braun, and K.-R. Müller. Kernel analysis of deep networks. *Journal of Machine Learning Research (JMLR)*, 12:2563–2581, 2011.

Y. Mroueh, S. Voinea, and T. A. Poggio. Learning with group invariant features: A kernel perspective. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

K. Muandet, K. Fukumizu, B. Sriperumbudur, B. Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends in Machine Learning*, 10(1-2):1–141, 2017.

A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *Journal of molecular biology*, 247(4):536–540, 1995.

R. M. Neal. *Bayesian learning for neural networks*. Springer, 1996.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust Stochastic Approximation Approach to Stochastic Programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

Y. Nesterov. *Introductory Lectures on Convex Optimization*. Springer, 2004.

G. Neu and L. Rosasco. Iterate averaging as regularization for stochastic gradient descent. In *Conference on Learning Theory (COLT)*, 2018.

B. Neyshabur, R. Tomioka, and N. Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory (COLT)*, 2015a.

B. Neyshabur, R. Tomioka, and N. Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015b.

B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro. The role of over-parametrization in generalization of neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

R. Novak, L. Xiao, Y. Bahri, J. Lee, G. Yang, J. Hron, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein. Bayesian deep convolutional networks with many channels are gaussian processes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

E. Oyallon and S. Mallat. Deep roto-translation scattering for object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

E. Oyallon, E. Belilovsky, and S. Zagoruyko. Scaling the scattering transform: Deep hybrid networks. In *International Conference on Computer Vision (ICCV)*, 2017.

M. Paulin, J. Revaud, Z. Harchaoui, F. Perronnin, and C. Schmid. Transformation pursuit for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

A. Pinkus. Approximation theory of the mlp model in neural networks. *Acta numerica*, 8:143–195, 1999.

A. Raghunathan, J. Steinhardt, and P. Liang. Certified defenses against adversarial examples. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

A. Raj, A. Kumar, Y. Mroueh, T. Fletcher, and B. Schoelkopf. Local group invariant representations via orbit embeddings. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

G. Raskutti, M. J. Wainwright, and B. Yu. Early stopping and non-parametric regression: an optimal data-dependent stopping rule. *Journal of Machine Learning Research (JMLR)*, 15(1):335–366, 2014.

H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

J. Rony, L. G. Hafemann, L. S. Oliveira, I. B. Ayed, R. Sabourin, and E. Granger. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

L. Rosasco, E. D. Vito, A. Caponnetto, M. Piana, and A. Verri. Are loss functions all the same? *Neural Computation*, 16(5):1063–1076, 2004.

F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

S. Rosset, G. Swirszcz, N. Srebro, and J. Zhu. ℓ_1 regularization in infinite dimensional feature spaces. In *Conference on Learning Theory (COLT)*, 2007.

K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann. Stabilizing training of generative adversarial networks through regularization. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann. Adversarially robust training through structured gradient regularization. *arXiv preprint arXiv:1805.08736*, 2018.

A. Rudi and L. Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pages 3215–3225, 2017.

A. Rudi, R. Camoriano, and L. Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

S. Saitoh. *Integral transforms, reproducing kernels and their applications*, volume 369. CRC Press, 1997.

H. Salman, G. Yang, J. Li, P. Zhang, H. Zhang, I. Razenshteyn, and S. Bubeck. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

P. Savarese, I. Evron, D. Soudry, and N. Srebro. How do infinite width bounded norm networks look in function space? In *Conference on Learning Theory (COLT)*, 2019.

R. E. Schapire and Y. Freund. *Boosting: Foundations and algorithms*. MIT Press, 2012.

R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The annals of statistics*, 26(5): 1651–1686, 1998.

L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Mądry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

M. Schmidt, N. Le Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1):83–112, 2017.

I. J. Schoenberg. Positive definite functions on spheres. *Duke Mathematical Journal*, 9 (1):96–108, 1942.

B. Schölkopf. *Support Vector Learning*. PhD thesis, Technischen Universität Berlin, 1997.

B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. 2001.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

H. Sedghi, V. Gupta, and P. M. Long. The singular values of convolutional layers. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

S. Shalev-Shwartz. SDCA without Duality, Regularization, and Individual Convexity. In *International Conference on Machine Learning (ICML)*, 2016.

S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research (JMLR)*, 14:567–599, 2013.

S. Shalev-Shwartz, O. Shamir, and K. Sridharan. Learning kernel-based halfspaces with the 0-1 loss. *SIAM Journal on Computing*, 40(6):1623–1646, 2011.

J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

L. Sifre and S. Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2013.

P. Y. Simard, Y. A. LeCun, J. S. Denker, and B. Victorri. Transformation invariance in pattern recognition–tangent distance and tangent propagation. In *Neural networks: tricks of the trade*, pages 239–274. Springer, 1998.

C.-J. Simon-Gabriel, Y. Ollivier, L. Bottou, B. Schölkopf, and D. Lopez-Paz. First-order adversarial vulnerability of neural networks and input dimension. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

A. Sinha, H. Namkoong, and J. Duchi. Certifying some distributional robustness with principled adversarial training. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

S. Smale and D.-X. Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1(01):17–41, 2003.

S. Smale, L. Rosasco, J. Bouvrie, A. Caponnetto, and T. Poggio. Mathematics of the neural response. *Foundations of Computational Mathematics*, 10(1):67–91, 2010.

A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2000.

A. J. Smola, Z. L. Ovari, and R. C. Williamson. Regularization with dot-product kernels. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

M. Soltanolkotabi, A. Javanmard, and J. D. Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2018.

D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research (JMLR)*, 19(1):2822–2878, 2018.

B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, G. R. Lanckriet, et al. On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6:1550–1599, 2012.

E. M. Stein. *Harmonic Analysis: Real-variable Methods, Orthogonality, and Oscillatory Integrals*. Princeton University Press, 1993.

I. Steinwart and A. Christmann. *Support vector machines*. Springer, 2008.

I. Steinwart, P. Thomann, and N. Schmid. Learning with hierarchical gaussian kernels. *preprint arXiv:1612.00824*, 2016.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

M. Telgarsky. Margins, shrinkage and boosting. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

A. Torralba and A. Oliva. Statistics of natural image categories. *Network: computation in neural systems*, 14(3):391–412, 2003.

A. Trouvé and L. Younes. Local geometry of deformable templates. *SIAM journal on mathematical analysis*, 37(1):17–59, 2005.

D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry. Robustness may be at odds with accuracy. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2008.

L. G. Valiant. A theory of the learnable. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 436–445. ACM, 1984.

M. J. van de Vijver et al. A Gene-Expression Signature as a Predictor of Survival in Breast Cancer. *New England Journal of Medicine*, 347(25):1999–2009, Dec. 2002.

L. van der Maaten, M. Chen, S. Tyree, and K. Q. Weinberger. Learning with marginalized corrupted features. In *International Conference on Machine Learning (ICML)*, 2013.

V. Vapnik. *The nature of statistical learning theory*. Springer, 2000.

V. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264, 1971.

J.-P. Vert and J. Mairal. Machine learning with kernel methods. Course in the "Mathématiques, Vision, Apprentissage" Master. ENS Cachan, 2017. URL `http://members.cbio.mines-paristech.fr/~jvert/svn/kernelcourse/slides/master2017/master2017.pdf`.

U. von Luxburg and O. Bousquet. Distance-based classification with lipschitz functions. *Journal of Machine Learning Research (JMLR)*, 5(Jun):669–695, 2004.

S. Wager, W. Fithian, S. Wang, and P. Liang. Altitude Training: Strong Bounds for Single-layer Dropout. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

G. Wahba. *Spline models for observational data*, volume 59. Siam, 1990.

M. J. Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

C. Wei, J. D. Lee, Q. Liu, and T. Ma. Regularization matters: Generalization and optimization of neural nets v.s. their induced kernel. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

T. Wiatowski and H. Bölcskei. A mathematical theory of deep convolutional neural networks for feature extraction. *IEEE Transactions on Information Theory*, 64(3): 1845–1866, 2018.

C. K. Williams. Computing with infinite networks. In *Advances in Neural Information Processing Systems (NIPS)*, 1997.

C. K. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

F. Williams, M. Trager, C. Silva, D. Panozzo, D. Zorin, and J. Bruna. Gradient dynamics of shallow low-dimensional relu networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

E. Wong and J. Z. Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research (JMLR)*, 11:2543–2596, 2010.

L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

B. Xie, Y. Liang, and L. Song. Diverse neural network learns true target functions. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

H. Xu, C. Caramanis, and S. Mannor. Robust regression and lasso. In *Advances in Neural Information Processing Systems (NIPS)*, 2009a.

H. Xu, C. Caramanis, and S. Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research (JMLR)*, 10(Jul):1485–1510, 2009b.

G. Yang. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*, 2019.

G. Yang and H. Salman. A fine-grained spectral perspective on neural networks. *arXiv preprint arXiv:1907.10599*, 2019.

Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

D. Yin, K. Ramchandran, and P. Bartlett. Rademacher complexity for adversarially robust generalization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

Y. Yoshida and T. Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.

S. Zagoruyko and N. Komodakis. Wide residual networks. 2016.

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017a.

C. Zhang, S. Bengio, and Y. Singer. Are all layers created equal? *arXiv preprint arXiv:1902.01996*, 2019.

K. Zhang, I. W. Tsang, and J. T. Kwok. Improved nyström low-rank approximation and error analysis. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

T. Zhang, B. Yu, et al. Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579, 2005.

Y. Zhang, J. D. Lee, and M. I. Jordan. ℓ_1-regularized neural networks are improperly learnable in polynomial time. In *International Conference on Machine Learning (ICML)*, 2016.

Y. Zhang, P. Liang, and M. J. Wainwright. Convexified convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2017b.

S. Zheng and J. T.-Y. Kwok. Lightweight stochastic optimization for minimizing finite sums with infinite data. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the robustness of deep neural networks via stability training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

D. Zou, Y. Cao, D. Zhou, and Q. Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. *Machine Learning*, 2019.