Weaving order from uncertainty: Design, Analysis, and Applications of Transport-based Generative Models

Biraj Pandey

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2024

Reading Committee:
Bamdad Hosseini, Chair
J. Nathan Kutz
Houman Owhadi

Program Authorized to Offer Degree:
Applied Mathematics

University of Washington

Abstract

Weaving order from uncertainty: Design, Analysis, and Applications of Transport-based Generative Models

Biraj Pandey

Chair of the Supervisory Committee:
Bamdad Hosseini
Department of Applied Mathematics

Generative machine learning algorithms are pivotal for advancing artificial intelligence and for gaining insights into biological neural systems. In this thesis, we present a comprehensive study that integrates theoretical analysis, efficient algorithm development, and biological applications in generative modeling.

We first establish a general theoretical framework for minimum divergence transport estimators, a prominent class of generative models. We derive a priori error bounds that quantify the generalization performance of these models in terms of model and sample complexity.

Building upon this theory, we introduce a flow-based transport algorithm for generative modeling that utilizes kernel methods to minimize Maximum Mean Discrepancy (MMD). Our method offers an efficient alternative to existing neural network approaches, achieving comparable performance with fewer parameters and reduced training time. We apply the theoretical results from the first part to derive generalization bounds for this algorithm.

Finally, we explore the intersection of generative modeling and neuroscience by developing a generative model for receptive fields in sensory neuronal systems using Gaussian processes. This model elucidates how sensory neurons transform inputs to create robust representations. Our biological insights inspire an initialization strategy that improves the efficiency of neural network training.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This thesis is the culmination of my doctoral research at the University of Washington. My deepest gratitude goes to my advisor, Bamdad Hosseini, whose brilliance, kindness, and unwavering support have been a guiding light throughout this journey. I am a better mathematician and, more importantly, a kinder human being because of your mentorship. I already know I will look back on our time working together with immense fondness and gratitude.

I am profoundly thankful to my family and friends, who have stood by me through every step of this endeavor. To my parents, Madhav and Gita, thank you for your boundless pride in my work; at times when I couldn't find any within myself, I found it reflected in you. To my brother, Bipul, your journey in science inspired me to start my own and gave me invaluable lessons to sustain it. To my sister-in-law, Khushboo, your unwavering optimism and encouragement have been an anchor for me. To my partner, Simran, your emotional support has been my steady ground through the toughest moments—I leaned on you more times than I can count.

I would also like to thank Alex, my academic brother, for sharing your wisdom on good coding practices, research organization, and for the countless hours of advice and camaraderie. Your support has been essential to this accomplishment.

Finally, I am grateful to the friends, colleagues, and peers I have had the privilege to meet—both within the department and across conferences. Your thoughtful discussions, patience, and curiosity have helped me refine my ideas, articulate my arguments, and situate my work within a broader scientific context.

To all of you: thank you for being part of this journey.

DEDICATION

To my parents Madhav and Gita, my brother Bipul, my sister-in-law Khushboo, and my partner Simran.

Chapter 1

INTRODUCTION

Generative modeling has emerged as a fundamental area in modern machine learning, offering a principled framework for understanding and recreating the patterns underlying complex datasets. By capturing the underlying probability distribution of a dataset, generative models empower a range of applications, from realistic image and video synthesis to advancing natural language processing, scientific simulations, and personalized medicine. The recent surge in generative modeling has pushed the boundaries of what can be achieved in domains such as art, healthcare, and fundamental science.

At the heart of generative modeling lies the mathematical task of learning a transformation $T : \mathbb{R}^d \to \mathbb{R}^d$ such that $T\#\eta \approx \nu$, where η is a simple reference measure, such as the standard Gaussian, and ν is a complex target measure that captures the data distribution. The goal is to construct T so that sampling $z \sim \eta$ and applying $T(z)$ produces samples distributed approximately as ν. Here, $T\#\eta$ denotes the pushforward measure, representing the transformed distribution obtained by applying T to samples from η.

A popular and theoretically grounded approach to generative modeling involves formulating the learning process as a minimum divergence transport (MDT) problem. Specifically, the objective is to solve:

$$\min_{T \in \mathcal{S}} D(T\#\eta, \nu),$$

where $\mathcal{S}$ is the hypothesis space of admissible transformations, and D is a divergence or discrepancy that quantifies the difference between the transformed measure $T\#\eta$ and the target measure ν. Examples of divergences D include the Kullback-Leibler (KL) divergence, the Wasserstein distance, and the maximum mean discrepancy (MMD), each tailored to empha-

size the specific properties of the distributions. The choice of $\mathcal{S}$ and D defines the modeling framework, with $\mathcal{S}$ often taking the form of parameterized neural networks, reproducing kernel Hilbert spaces (RKHS), or other structured function classes.

MDT models encompass a range of influential generative modeling paradigms. Seminal works in this area include the theory of optimal transport [219, 170], which minimizes the Wasserstein distance, the development of normalizing flows [181, 162], which parameterize T as an invertible transformation that minimize KL Divergence, and neural ODE-based approaches [47, 94], where T is defined via the solution of an ordinary differential equation.

While generative modeling has achieved remarkable empirical success, understanding these models mathematically remains a critical challenge. Such understanding involves analyzing the behavior of generative models before training, providing *a priori* guarantees that can guide their design and analysis. Theoretical tools such as generalization bounds and approximation theory play a vital role in this endeavor. Recent advances in this area include works on the generalization theory of GANs [131, 212], the sample complexity of optimal transport [227, 199], and approximation properties of kernel and neural network-based models [37, 176]. These studies help to elucidate the factors that influence the performance of generative models, such as model capacity, regularization, and the number of training samples.

This dissertation contributes to this growing body of work by exploring the theory, algorithm development, and applications of generative modeling. Generative models serve as the unifying theme, linking different approaches and perspectives. The contributions of this thesis include: (1) the development of a novel transport-based generative model, (2) the derivation of theoretical results that provide insights into the behavior and limitations of transport-based models, and (3) the application of generative modeling concepts to neuroscience, where they provide a framework for understanding sensory encoding.

1.1 Overview and Organization of the Dissertation

The dissertation is organized into three main chapters, each addressing a critical aspect of generative modeling while maintaining a cohesive narrative around the central theme of understanding, improving, and applying generative models.

In Chapter 2, we introduce a new transport-based generative model where velocity fields of the flow lie in a Reproducing Kernel Hilbert Space (RKHS). Our choice of RKHS is motivated by their mathematical simplicity and rich history in analysis and algorithm development. The main contributions include theoretical foundations, error bounds that balance approximation, regularization, and sample complexity, and numerical experiments demonstrating the model's competitiveness to state-of-the-art models in several standard benchmark examples.

In Chapter 3, we establish a framework for deriving quantitative generalization error rates for minimum divergence transport (MDT) models. The core contributions are master theorems that decompose the generalization error into approximation error and stochastic error arising from sample variance. To demonstrate the framework's utility, we apply it to concrete examples, including GANs, RKHS-based transport maps, and polynomial maps. Finally, we assess the validity and sharpness of our bounds through numerical experiments, comparing theoretical predictions with empirical results.

In Chapter 4, we apply generative modeling to the study of sensory encoding in neuroscience, inspired by the parallels between biological neural systems and artificial networks. The central idea is to model the receptive fields of sensory neurons as random samples from a Gaussian process with a carefully chosen covariance structure. The core contributions include demonstrating that these structured receptive fields closely align with experimental data from mechanosensory and visual neurons, providing realistic and biologically plausible models of sensory encoding. Furthermore, we show that insights from this model enhance learning efficiency in artificial networks.

Chapters 2 through 4 start with an introductory section titled "Prelude to the article"

that summarizes the main results of each article and highlights their connection to the rest of this document. Otherwise, the content of each of these chapters is identical to the published version of the articles, except for minor corrections and changes to the numbering of sections, equations, etc.

1.2 Contribution of authors

- (Chapter 2) Submitted to *SIAM Journal on the Mathematics of Data Science* [159].
- (Chapter 3) In preparation.
- (Chapter 4) Published in *PLOS Computational Biology* [160]

Chapter 2

DIFFEOMORPHIC MEASURE MATCHING WITH KERNELS FOR GENERATIVE MODELING

2.1 Prelude to the chapter

In this chapter, we develop a generative model based on the flow of an Ordinary Differential Equation (ODE) parameterized using a Reproducing Kernel Hilbert Space (RKHS) model. Our aim is to construct algorithms that not only perform efficiently in practice but also come with rigorous theoretical guarantees. Specifically, we seek to quantitatively understand how the approximation error of the model depends on its complexity and the number of training data samples.

The dominant paradigm in generative modeling is *minimum divergence measure transport* [140]. The central idea is to learn a transport map $T : \mathbb{R}^d \to \mathbb{R}^d$ that pushes forward samples from an easy-to-simulate reference measure η (e.g., a standard normal distribution) to samples distributed according to a complex target measure ν. This map is learned by minimizing a discrepancy or metric D between $T\#\eta$ and ν, where common choices for D include the Kullback-Leibler divergence [30] and Maximum Mean Discrepancy (MMD) [95].

Modern generative models often parameterize the transport map as the flow of an ODE whose velocity field is represented by a neural network [47, 94]. Mathematically, given a velocity field $v(t, x)$, the flow $\phi(t, x)$ of an ODE is defined by the equation,

$$\frac{d\phi}{dt} = v(t, \phi(t, x)), \quad \phi(0, x) = x,$$

where t denotes time and x represents the initial data or reference sample. The solution at $\phi(T, x)$ at some arbitrary time $t = T$ serves as the transport map $T(x)$, pushing forward samples from the reference measure to the target measure. Despite their impressive empir-

ical performance, obtaining quantitative error rates for such neural network-based models remains challenging.

In our work, we constrain the velocity field $v(t, x)$ to lie within an RKHS, i.e., a Hilbert space of functions where point-wise evaluation is a bounded linear functional [197]. To reduce the discrepancy between the transported measure and the target measure, we minimize the MMD which is a statistical distance between probability distributions defined by embedding the distributions in an RKHS [95, 147]. Our approach builds upon prior work in diffeomorphic matching and kernel methods [234, 85, 156]. By framing the generative modeling problem within an RKHS context, we leverage the theoretical advantages of kernel methods, which have been extensively studied in machine learning and statistics [7, 197, 228].

In Section 2.3, we present an error analysis of our generative model, establishing bounds that quantify how the quality of our approximation depends on various factors such as model complexity, sample size, and potential model misspecification. Specifically, our analysis reveals the effects of bias and variance on the model. The bias arises from the approximation error due to the finite representation in the RKHS, while the variance stems from the finite number of training samples used to estimate the distributions. This error decomposition provides insights into how increasing model complexity or sample size can reduce the respective errors, guiding the design of efficient and accurate generative models.

In Section 2.4, we describe the numerical implementation of the algorithm in detail, specifying the choice of model parametrization and strategies for tuning the hyperparameters. In Section 2.5, we demonstrate several applications of our algorithm in generative modeling and parameter inference. The generative modeling problems range from two-dimensional toy datasets to high-dimensional real-world datasets derived from neutrino experiments and electricity consumption [163]. In the parameter inference task, we identify the parameters of a Lotka-Volterra ODE system in a likelihood-free fashion [13], showcasing the versatility and effectiveness of our approach.

The contents of this chapter appear in the article [159] and is under revision in the SIAM Journal on the Mathematics of Data Science (SIMODS).

2.2 Introduction

Sampling/simulation of probability measures is a fundamental task in data science, computational statistics, and uncertainty quantification (UQ). Given a *target* probability measure $\nu \in \mathbb{P}(\mathbb{R}^d)$ for some $d \geq 1$, the goal of sampling is to simulate a random variable that is distributed according to this target. Markov chain Monte Carlo (MCMC) [185, 209, 54, 97, 55, 25, 26, 64, 104, 82, 105] and variational inference (VI) [30, 241, 75] algorithms are perhaps the most well-known families of algorithms for this task, but in recent years, there has been a significant increase in interest surrounding the family of transport-based sampling algorithms [140] due to the impressive performance of generative models [149, 115] such as generative adversarial networks (GANs) [90, 89] and normalizing flows (NFs) [181, 123, 162]. Broadly speaking, these transport-based algorithms find a transport map $T^\star$ such that $T^\star \# \eta \approx \nu$ for some reference measure $\eta \in \mathbb{P}(\mathbb{R}^d)$[1] that is easy to simulate, e.g., a standard normal. Then, samples from the target ν are (approximately) simulated by drawing $z \sim \eta$ and evaluating $T^\star(z)$. The map $T^\star$ is a minimum divergence estimator [165], often found by solving problems of the form

$$T^\star := \operatorname*{arg\,min}_{T \in \mathcal{T}} D(T\#\eta, \nu) + \lambda_R R(T), \tag{2.1}$$

where $D : \mathbb{P}(\mathbb{R}^d) \times \mathbb{P}(\mathbb{R}^d) \to [0, +\infty]$ is a statistical divergence or loss, $\mathcal{T}$ is an appropriate, often parametric, family of transport maps from $\mathbb{R}^d$ to $\mathbb{R}^d$, and $R : \mathcal{T} \to [0, +\infty]$ is a regularization/penalty term with parameter $\lambda_R \geq 0$ that provides additional stability.

In this chapter, we are particularly interested in the setting where $\mathcal{T}$ is the space of maps given by the flow of an ODE, as popularized in [47, 94], i.e.,

$$\mathcal{T} := \left\{ T : \mathbb{R}^d \to \mathbb{R}^d \;\middle|\; T(x; v) = \phi(1, x), \quad \phi_t = v(t, \phi), \quad t \in (0, 1], \quad \phi(0, x) = x, \quad v \in \boldsymbol{\mathcal{Q}} \right\}, \tag{2.2}$$

where $\boldsymbol{\mathcal{Q}}$ is a family of vector fields, often taken to be a neural network (NN) class in modern generative modeling [47, 94, 154]. In this chapter, we consider a family of such dynamic

[1] In practice it is customary to define η on a lower dimensional latent space, but we will not consider this setting in this article.

transport maps based on the theory of kernel methods and present theoretical analysis and benchmark experiments for our method. In the rest of this section, we give a summary of our contributions in Section 2.2.1 followed by a review of relevant literature in Section 2.2.2, and an outline of the chapter in Section 2.2.3.

2.2.1 Summary of contributions

Below, we will give a concise summary of our mathematical formulation and main results with minimal definitions, and refer the reader to Section 2.3.1 for a review of the relevant RKHS theory. Drawing inspiration from previous works in image registration and diffeomorphic matching [85, 234, 69, 156, 58] we take $\boldsymbol{\mathcal{Q}}$ to be a vector valued Reproducing Kernel Hilbert Space (RKHS) [118] on the set $\Gamma := [0,1]\times\mathbb{R}^d$ and consider the following instance of Eq (2.1):

$$\begin{cases} \underset{v\in\boldsymbol{\mathcal{Q}}}{\text{minimize}} & D(\phi(1,\cdot)\#\eta^N, \nu^N) + \lambda_R \|v\|^2_{\boldsymbol{\mathcal{Q}}}, \\ \text{subject to (s.t.)} & \phi_t = v(t,\phi), \quad \phi(0,x) = x, \end{cases} \tag{2.3}$$

where we replaced the reference η and the target ν with their empirical approximations η^N and ν^N obtained from N-i.i.d. samples. Towards the design of a practical algorithm, we take D to be the *maximum mean discrepancy (MMD)* defined by a Mercer kernel $K : \mathbb{R}^d\times\mathbb{R}^d \to \mathbb{R}$, i.e., it is symmetric, positive, and separately continuous. Following the definition Eq (2.7) this divergence takes the simple form

$$\mathrm{MMD}^2_K(\phi(1,\cdot)\#\eta^N, \nu^N) = \frac{1}{N^2}\left(\sum_{i,j=1}^N K(\phi(1,x_i),\phi(1,x_j)) + \sum_{i,j=1}^N K(y_i,y_j) - 2\sum_{i,j=1}^N K(\phi(1,x_i),y_j)\right) \tag{2.4}$$

where we used $\{x_i\}_{i=1}^N$ and $\{y_i\}_{i=1}^N$ to denote i.i.d. samples from η and ν respectively. We further take $\boldsymbol{\mathcal{Q}}$ to be the RKHS of a matrix-valued kernel $\boldsymbol{Q} : \Gamma\times\Gamma \to \mathbb{R}^{d\times d}$ of the diagonal form $\boldsymbol{Q}(s,s') = V(s,s')I$ where $I \in \mathbb{R}^{d\times d}$ is the identity matrix and $V : \Gamma\times\Gamma \to \mathbb{R}$ is a second Mercer kernel, independent of K. Finally, we consider a set of *inducing points* $S := \{s_1,\ldots,s_M\} \subset \Gamma$ [214, 223, 65]

and approximate Eq (2.3) with the discrete problem

$$\begin{cases} \underset{\{c_\ell\}_{\ell=1}^d \subset \mathbb{R}^{M\times d}}{\text{minimize}} & \mathrm{MMD}_K(\phi(1,\cdot)\#\eta^N, \nu^N) + \lambda_R \sum_{\ell=1}^d c_\ell^T V(S,S) c_\ell, \\ \text{s. t.} & \phi_t = v(t,\phi), \quad \phi(0,x) = x, \\ & v(s) = (v_1(s), \ldots, v_d(s)), \quad v_\ell(s) = c_\ell^T V(S,s), \end{cases} \tag{2.5}$$

where $V(S,S) \in \mathbb{R}^{M\times M}$ denotes the kernel matrix with entries $(V(S,S))_{ij} = V(s_i, s_j)$ and $V(S,s)$ is a (column) vector field with entries $(V(S,s))_i = V(s_i, s)$. The coefficient vectors $\{c_\ell\}_{\ell=1}^d \subset \mathbb{R}^M$ denote the parameters of our model and are the main target of training in Eq (2.5). We also note that once the requisite kernels K, V, the regularization parameter λ, and the set of inducing points S are chosen, then the above formulation can be implemented using an off-the-shelf ODE solver to approximate the flow $\phi(1,\cdot)$.

Our main contributions are the theoretical analysis, the algorithmic development and the benchmarking of the proposed solution Eq (2.5) for sampling and inference.

For our theoretical analysis we primarily consider the setting where the reference η and target ν are compactly supported and consider the closely related problem

$$\begin{cases} \underset{\{c_\ell\}_{\ell=1}^d \subset \mathbb{R}^{M}}{\text{minimize}} & \mathrm{MMD}_K(\phi(1,\cdot)\#\eta^N, \nu^N) \\ \text{s. t.} & \phi_t = v(t,\phi), \;\; \phi(0,x) = x, \;\; v(s) = (v_1(s), \ldots, v_d(s)), \\ & v_\ell(s) = c_\ell^T V(S,s), \;\; \sum_{\ell=1}^d c_\ell^T V(S,S) c_\ell \le r^2, \end{cases} \tag{2.6}$$

for some $r^2 > 0$. The above problem is closely related to Eq (2.5), indeed the latter can be obtained by using a Lagrange multiplier to relax the inequality constraints and leads to more efficient algorithms; see Section 2.3.3 for more details on how the two minimizers are related. We then obtain the following theorem that gives a quantitative characterization of the approximation and generalization errors of the transport map obtained by solving Eq (2.6):

Main Theorem 1. *Let $\Omega \subset \mathbb{R}^d$ be bounded and let $\Gamma = [0,1] \times \Omega$. Suppose $K : \Omega \times \Omega \to \mathbb{R}_{\ge 0}$ is a Mercer kernel with RKHS $\mathcal{K}$ such that $\sup_{x,x' \in \Omega} \frac{\|K(x,\cdot) - K(x',\cdot)\|_{\mathcal{K}}}{|x-x'|} < +\infty$. Suppose $\eta, \nu \in \mathbb{P}(\Omega)$, and let $V : \Gamma \times \Gamma \to \mathbb{R}$ be a Mercer kernel such that $V \in C^{2k}(\Gamma \times \Gamma)$ and that elements of its RKHS $\mathcal{V}$ vanish at the boundary of Ω. Suppose there exists a Lipschitz vector field $v^\dagger$ such that $T(\cdot; v^\dagger)\#\eta = \nu$*

(recall the notation of Equation (2.2)). For some $r > 0$, let $T(\cdot; v_r^{S,N})$ be the transport map defined by solving Eq (2.6) and define $h_S := \sup_{s\in\Gamma} \inf_{s'\in S} |s - s'|$ to be the fill-distance of $S \subset \Gamma$. Then, if $h_S > 0$ is sufficiently small it holds with probability $1 - \delta$, for $\delta > 0$, that

$$\mathrm{MMD}_K(T(\cdot; v_r^{S,N})\#\eta, \nu) \leq C_2 \Bigg[\big(\exp(C_1 r) - 1\big) h_S^k + \sqrt{\frac{1}{N}} \left(2 + \sqrt{\log\left(\frac{1}{\delta}\right)}\right) + \frac{\exp(L_{v^\dagger}) - 1}{L_{v^\dagger}} \inf_{v\in\boldsymbol{\mathcal{Q}}_r} \|v - v^\dagger\|_\infty \Bigg],$$

where $C_1, C_2 > 0$ are constants independent of S, N, and $v^\dagger$, $L_{v^\dagger} := \sup_{s,s'\in\Gamma} \frac{|v^\dagger(s) - v^\dagger(s')|}{|s-s'|}$ is the Lipschitz constant of $v^\dagger$, and $\boldsymbol{\mathcal{Q}}_r := \{v : \Gamma \to \mathbb{R}^d \mid \sum_{\ell=1}^d \|v_\ell\|_{\mathcal{V}}^2 \leq r^2\}$.

A complete proof of this theorem, including extensions and other useful auxiliary results, is given in Section 2.3. The above result gives a complete picture of the approximation bias, sample complexity of our algorithm, and model misspecification error. In the context of generative models, and more broadly, of computational transport, such quantitative error bounds that account for both approximation errors and sample complexity are very challenging to obtain; see for example [108, 63]. We note that the above bound does not take into account the approximation error of the ODE solver Eq (2.6) or the optimization gap in finding a global minimizer.

Our bound is interesting in various aspects: (1) The first term, involving the fill-distance h_S, controls the approximation error of the model and reflects the smoothness of the underlying RKHS $\mathcal{V}$ and the complexity of our model class. Indeed, this bound matches classical rates for scattered data approximation with kernel methods [228]; (2) The second term is a generalization bound for minimum MMD estimators following [37] and matches the minimax optimal approximation rate of kernel mean embeddings of measures [204, 215]; (3) The third term can be viewed as a model misspecification error, i.e., choosing a kernel V such that the resulting RKHS does not include the ground truth vector field $v^\dagger$ or that r is simply too small. Thus, if the model is well-specified, i.e., $v^\dagger \in \boldsymbol{\mathcal{Q}}_r$, then this term will simply vanish. Of course, there is a trade-off here since the constant in the first term blows up exponentially with r. Finally, we note that while the second term in our bound is dimension independent, the first term is effectively dependent on the dimension of the problem since the fill-distance h_S scales as $M^{-1/d}$ [182, Eqn. 1.2], recall that M is the number of inducing points. So, to ensure that h_S is sufficiently small one needs an increasingly large

set of inducing points, however, the resulting approximation error still scales as $M^{-k/d}$ which is acceptable when $k > d$, i.e., the class $\mathcal{V}$ is sufficiently smooth. However, there is a trade-off involving the smoothness of $\mathcal{V}$. Smoother $\mathcal{V}$ (with large k) can reduce the approximation error term but might fail to adequately model non-smooth velocity fields, increasing the model misspecification error.

For our numerical results we solve Eq (2.5) using off-the-shelf ODE solvers and stochastic gradient descent; see Section 2.4 for details. We refer to our algorithm as KODE (Kernel ODE transport) and benchmark it on various data sets in low- or high-dimensional settings; on occasion, we compare KODE with the OT-Flow algorithm of [154], which is a neural net method based on the dynamic formulation of the optimal transport (OT) problem. Figure 2.1 shows an example of our results for a set of 2D benchmarks of different complexity where we (forward) transport a standard Gaussian reference η to a complicated target ν. The bottom row of this figure shows the backward transport/normalizing flow problem of pulling ν back to η simply by running the ODE backward without any training, revealing the efficacy of our method. We also present extensive studies on the effects of hyper-parameters and the choice of kernels and RKHS norms.

Finally, we also present a simple modification of our formulation that enables KODE to perform likelihood-free/amortized inference, i.e., we transport η to ν using additional constraints leading to a triangular transport map which we call Triangular KODE (T-KODE), based on the theory of triangular transport of measures [13]. The resulting model is only trained once but it can identify arbitrary conditional measures of ν along pre-specified coordinates as demonstrated in Figure 2.2. Details and additional benchmarks for this method are collected in Section 2.5.4.

2.2.2 Relevant Literature

Below, we give a summary of the relevant literature to our work. Since the literature on generative modeling and transport is vast we keep the discussion focused on works that are closely related to ours and give a few references for broader topics.

Diffeomorphic matching and learning

The early developments in matching measures originated from the field of diffeomorphic matching or learning, which aims to align different data sets by transforming them into a common coordinate

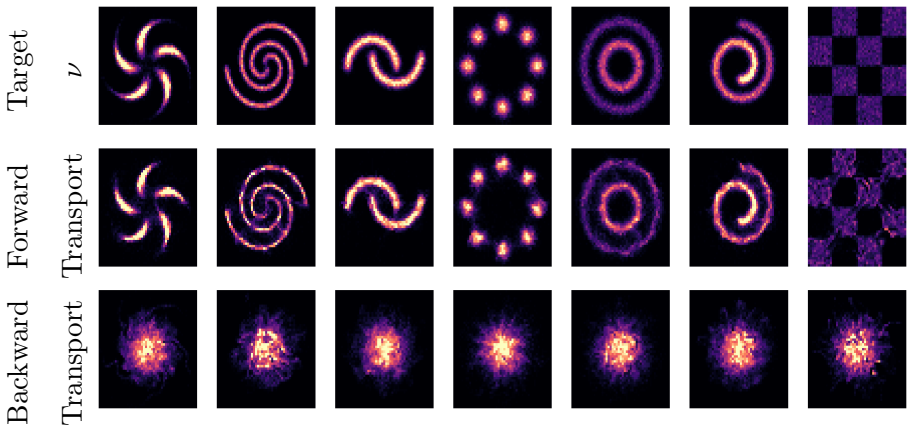

Figure 2.1: **Transport experiments on 2D benchmarks using KODE**: (Top row) The empirical samples from the target ν; (Middle row) Samples generated by transporting a standard Gaussian reference η; (Bottom row) Samples generated by transporting ν backward towards η.

system [245, 21, 117]. An example of an application includes aligning the images of the same scene captured from various viewpoints, sensors, and times of day. Computational algorithms for diffeomorphic matching shares a lot of similarities with our proposed method [234, 236] as well as many recent techniques in generative modeling such as continuous NFs [94, 123]. Indeed, the problem of matching measures and distributions has been studied extensively in [20, 69, 70, 235] although these works were mainly focused on matching of images and shapes in 2D or 3D as opposed to sampling of high-dimensional measures.

In diffeomorphic matching, flows of ODEs are utilized as diffeomorphic maps to continuously deform one image or volume into another. Notably, these diffeomorphic maps are often chosen to be flows of vector fields belonging to Reproducing Kernel Hilbert Spaces (RKHS) [234, 156, 58], distinguishing them from current methods in machine learning that primarily use neural network parameterizations. In this light, our proposed methodology also falls within the category of diffeomorphic matching with a particular focus towards sampling (possibly) high-dimensional probability measures.

Closely related approaches to ours in this domain include [85], which was a major inspiration

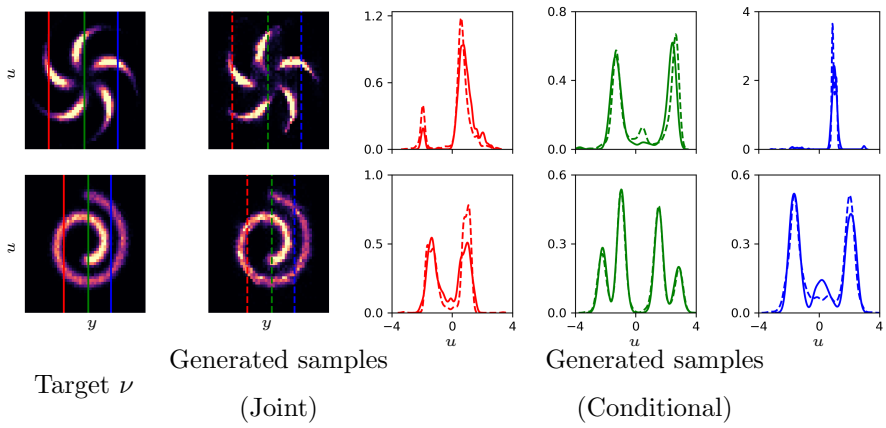

Figure 2.2: **Conditional sampling using Triangular KODE for two-dimensional benchmarks**: The left-most column shows the target measure ν while the red, green, and blue lines denote slices along which conditional samples are generated. The next column shows generated samples by KODE. The remaining panels compare ground truth (solid lines) and generated (dashed lines) kernel density estimators of the requisite conditional distributions using triangular KODE.

for us and employs a formulation that is a subset of our framework. Also, the theoretical analysis in that work is limited to existence and consistency results as opposed to our quantitative rates. The recent article [58] also proposes a similar formulation to ours, with the main differences being the choice of Sinkhorn divergences in place of MMD and the focus on 2D or 3D examples. Another inspiration for our work is [156], which connects modern techniques, such as residual neural nets to the formulation of algorithms for diffeomorphic matching, although that article is primarily focused on supervised learning as opposed to generative modeling. Another important point of departure for us from the aforementioned works is that they utilize geodesic shooting [236, 156] to compute their transport maps as opposed to direct minimization in our setting. Exploring the geodesic shooting method in our framework is interesting but non-trivial, since the derivation of that method relies on the assumption that the RKHS norm of v is of the form $\|v\|^2_{\boldsymbol{\mathcal{Q}}} = \int_0^1 \|v(t,\cdot)\|^2_{\boldsymbol{\mathcal{Q}}'} dt$ where $\boldsymbol{\mathcal{Q}}'$ is an RKHS on Ω. Our experiments in Section 2.5 suggest that imposing additional smoothness in time results in better performance and smoother transport maps. Alternatively, one can also enforce

additional smoothness by penalizing Riemannian derivatives of the velocity fields, as explored in [217, 83]. Finally, we mention the master's thesis [180] that gives an excellent overview of the connection between kernel ODEs and various Neural ODE models from the perspective of optimal control, including applications to measure transport.

ODE models in machine learning

The past decade has seen an explosion in the research and development of increasingly expressive generative models, specially in the context of image generation, going back to the introduction of GANs in [90]. Since then, numerous extensions of GANs [6, 96, 28, 129] have been proposed, followed by new families of generative models such as NFs [181, 162, 123] and more recently diffusion models [203, 42] and stochastic interpolants [3, 2]. A core idea in the aforementioned flow models is the parameterization of transport maps via the compositions of parametric maps that are simple and, often, invertible. In the case of normalizing flows, these constraints were partially motivated by the use of the Kullback-Liebler (KL) divergence as a training loss that takes a particularly simple form during training for the normalizing flow direction, i.e., pulling the target ν to the reference η. Then, generating new samples from ν requires the inversion of the resulting flow in an efficient manner. This line of thinking led to considerable interest in continuous time dynamic models, broadly referred to as neural ODEs [190, 47, 94]; simply put, a model akin to Eq (2.3) with v parameterized via a neural net.

The analysis of neural ODE models has attracted some attention in the literature although most of the existing works focus on the proof of universal approximation results for diffeomorphic maps [130, 112] and do not provide a quantitative approximation rates as we do in Main Theorem 1 and do not consider generalization bounds. To our knowledge, [141] presents the most comprehensive analysis of neural ODEs from the perspective of transport problems, focusing on generalization bounds for density estimation, as opposed to the sampling/generative modeling in our case. The results of that work combine generalization bounds and approximation errors for neural ODE models, and in that sense, are very similar to our main theorem, however: (1) their analysis is limited to the normalizing flow problem with the KL loss; (2) it is primarily focused on neural network models; and (3) concerns density estimation as opposed to sampling in our case.

We also note that there is a growing literature on the applied analysis of transport problems as it pertains to generative modeling for non-ODE models, such as [14] which proposes master theorems for characterizing the approximation error of transport maps for sampling that we utilize in the proof of our main results. The articles [239, 240] consider sparse polynomial approximations of triangular transport maps; a particularly useful construction in the context of normalizing flows. While the aforementioned articles are primarily concerned with the approximation errors of parameterized transport maps, the articles [111, 224] study the statistical consistency and sample complexity of such problems as it pertains to triangular maps.

Connections to OT

The theory of OT is now a mature field of mathematics [220, 193] with deep connections to probability theory, partial differential equations, and statistics. In recent years, there has been a growing interest in the computational aspects of optimal transport [170] both in the development of algorithms for computing optimal maps [56] as well as their applications in fields such as machine learning, data science [6, 146, 127, 72, 244], biology [195], medicine [93], shape analysis [210], and economics [80]. The wide adoption of OT in practice has also led to growing interest in the applied analysis of OT problems and, in particular, OT maps [108, 63, 59, 60, 172]. Our formulation of KODE resembles the dynamic formulation of OT, often known as the Benamou-Brenier formulation [220, pp. 159] which is also the basis of the OT-Flow algorithm [154], but it is different from that problem in two important aspects: (1) our dynamics do not match the target measure ν exactly (we only aim to minimize the MMD) while the Benamou-Brenier formulation aims to push the reference η to the target ν exactly; (2) we regularize our velocity fields in an RKHS in both space and time, while Benamou-Brenier minimizes a Bochner integral.

2.2.3 Outline

The rest of the article is organized as follows: Section 2.3 contains our theoretical analysis of the KODE methodology and in particular, the proof of Main Theorem 1. Details of our algorithms and numerical implementations are collected in Section 2.4 followed by numerical results and experiments in Section 2.5. We conclude the article in Section 2.6 with a summary of remaining open

questions and future directions.

2.3 Theory

We dedicate this section to the theoretical analysis of problem Eq (2.5), proving a series of auxiliary results that give Main Theorem 1 as a corollary but also generalize the setting of that result.

2.3.1 Brief review of scalar and vector valued RKHSs and MMD

We begin with a brief review of kernel methods in the classic setting of real-valued kernels as well as operator/matrix-valued kernels since we need both for our theoretical exposition. For brevity, we will not give an exhaustive treatment of these topics and only review the material needed in the paper. The interested reader may refer to [24, 157] for standard RKHS theory; [4, 118, 156] for operator/matrix-valued kernels; and [147] for a review of kernel mean embeddings and MMD.

For a domain $\Omega \subseteq \mathbb{R}^d$ a function $K : \Omega \times \Omega \to \mathbb{R}$ is called a symmetric and positive definite kernel on Ω if $K(x, x') = K(x', x)$, $\forall x, x' \in \Omega$ and for any collection of points $X := \{x_1, \dots, x_M\} \subset \Omega$ the kernel matrix $K(X, X) \in \mathbb{R}^{M \times M}$, with entries $K(X, X)_{ij} = K(x_i, x_j)$, is positive definite. We say that the kernel K is *Mercer* if it is separately continuous (i.e., continuous in each of its inputs) in addition to being symmetric and positive definite. Associated to the kernel K is an RKHS $\mathcal{K}$ with inner product $\langle \cdot, \cdot \rangle_{\mathcal{K}}$ and norm $\| \cdot \|_{\mathcal{K}}$ satisfying the reproducing property $f(x) = \langle f, K(x, \cdot) \rangle_{\mathcal{K}}$ for all $f \in \mathcal{K}$ and $x \in \Omega$. Let us introduce the shorthand notation $K(X, x) := (K(x_1, x), \dots, K(x_M, x)) \in \mathcal{K}^M$ as a *column vector field*, similarly $K(x, X)$ for the analogous *row vector field*. Then, functions of the form $f(x) = c^T K(X, x)$ for a (column) vector $c \in \mathbb{R}^M$ naturally belong to $\mathcal{K}$ and, by the reproducing property, we have the useful identity $\|f\|_{\mathcal{K}}^2 = c^T K(X, X)c$. For a second function $g(x) = (c')^T K(X', x)$, with a second point cloud $X' = \{x'_1, \dots, x'_{M'}\} \subset \Omega$ and vector of coefficients $c' \in \mathbb{R}^{M'}$, we also have the identity $\langle f, g \rangle_{\mathcal{K}} = c^T K(X, X')c'$.

Given the Mercer kernel K we also define the space $\mathbb{P}_K(\Omega) \subset \mathbb{P}(\Omega)$ of Borel probability measures μ for which $\int_\Omega \sqrt{K(x, x)}\mu(\mathrm{d}x) < +\infty$ along with the kernel mean embedding $I_K : \mathbb{P}_K(\Omega) \to \mathcal{K}$ where $I_K(\mu) := \int_\Omega K(x, \cdot)\mu(\mathrm{d}x)$. With this notation at hand, we can now introduce the MMD, as

a discrepancy defined on the space $\mathbb{P}_K(\Omega)$:

$$\begin{aligned}\mathrm{MMD}_K(\rho_1,\rho_2) := \|I_K(\rho_1) - I_K(\rho_2)\|_{\mathcal{K}} \equiv \Bigg(&\int_{\Omega\times\Omega} K(x,x')\rho_1(\mathrm{d}x)\rho_1(\mathrm{d}x') \\ &+ \int_{\Omega\times\Omega} K(x,x')\rho_2(\mathrm{d}x)\rho_2(\mathrm{d}x') - 2\int_{\Omega\times\Omega} K(x,x')\rho_1(\mathrm{d}x)\rho_2(\mathrm{d}x') \Bigg)^{1/2},\end{aligned} \tag{2.7}$$

We can naturally extend the MMD to all of $\mathbb{P}(\Omega)$ by setting its value to ∞ if either of the input measures do not belong to $\mathbb{P}_K(\Omega)$. We say the kernel K is *universal* if $\mathrm{MMD}_K(\rho_1,\rho_2) = 0$ implies that $\rho_1 = \rho_2$, in which case the MMD becomes a divergence in the parlance of [29]. Finally, observe that by taking ρ_1, ρ_2 in the above formula to be empirical measures, for example $\rho_1 = \frac{1}{M}\sum_{i=1}^{M}\delta_{x_i}$ and $\rho_2 = \frac{1}{M'}\sum_{i=1}^{M'}\delta_{x'_i}$ for $\{x_i\}_{i=1}^{M}, \{x'_i\}_{j=1}^{M'} \subset \Omega$, yields the familiar expression

$$\mathrm{MMD}_K(\rho_1,\rho_2) = \left(\frac{1}{M^2}\sum_{i,j=1}^{N} K(x_i,x_j) + \frac{1}{M'^2}\sum_{i,j=1}^{M} K(x'_i,x'_j) - \frac{2}{MM'}\sum_{i=1}^{M}\sum_{j=1}^{M'} K(x_i,x'_j) \right)^{1/2},$$

which gives Eq (2.4) in our introductory formulation of KODE.

Analogously to the case of real-valued RKHSs, we say that a matrix/operator-valued kernel $\boldsymbol{Q} : \Omega \times \Omega \to \mathbb{R}^{d\times d}$ is Mercer if it is separately continuous, symmetric and positive definite, i.e., $\boldsymbol{Q}(x,x') = \boldsymbol{Q}(x',x)^T$ and for any set of points $X = \{x_1,\ldots,x_M\} \subset \Omega$ and block-vector $Y = [y_1,\ldots,y_M] \in \mathbb{R}^{d\times M}$ [2] it holds that $Y^T\boldsymbol{Q}(X,X)Y := \sum_{i,j=1}^{M} y_i^T\boldsymbol{Q}(x_i,x_j)y_j \geq 0$ where we introduced our compressed notation for the multiplication of block-vectors with a matrix. Every Mercer kernel $\boldsymbol{Q}$ is in one-to-one correspondence with a (vector-valued) RKHS $\boldsymbol{\mathcal{Q}}$ of functions $q : \Omega \to \mathbb{R}^d$ equipped with the inner product $\langle\cdot,\cdot\rangle_{\boldsymbol{\mathcal{Q}}}$ and norm $\|\cdot\|_{\boldsymbol{\mathcal{Q}}}$ satisfying the reproducing property: $\langle q, \boldsymbol{Q}(x,\cdot)\rangle_{\boldsymbol{\mathcal{Q}}} = q(x)$. For functions of the form $q(x) = \sum_{j=1}^{M} c_j^T\boldsymbol{Q}(x_j,x)$ and $q'(x) = \sum_{j=1}^{M'} c_j'^T\boldsymbol{Q}(x'_j,x)$ with column coefficient vectors $\{c_j\}_{j=1}^{M}, \{c'_j\}_{j=1}^{M'} \subset \mathbb{R}^d$. Further introduce the block-vectors $C = [c_1,\ldots,c_M] \in \mathbb{R}^{d\times M}$ and $C' = [c'_1,\ldots,c'_{M'}] \in \mathbb{R}^{d\times M'}$ and point clouds $X = \{x_1,\ldots,x_M\} \subset \Omega$ and $X' = \{x'_1,\ldots,x'_{M'}\} \subset \Omega$ as before. We then have the identity $\langle q, q'\rangle_{\boldsymbol{\mathcal{Q}}} := C^T\boldsymbol{Q}(X,X')C' = \sum_{i=1}^{M}\sum_{j=1}^{M'} c_i^T\boldsymbol{Q}(x_i,x'_j)c'_j$ and subsequently $\|q\|^2_{\boldsymbol{\mathcal{Q}}} = C^T\boldsymbol{Q}(X,X)C$. Of particular importance to our exposition is the family of diagonal matrix-valued kernels of the form $\boldsymbol{V}(x,x') = V(x,x')I$ with associated RKHS $\boldsymbol{\mathcal{V}}$ where $I \in \mathbb{R}^{d\times d}$ is the identity matrix and

[2] One can think of Y as a matrix, but it is more helpful in this context to consider it as a column vector of size M whose entries are in $\mathbb{R}^d$, i.e., a block-vector.

$V : \Omega \times \Omega \to \mathbb{R}$ is a scalar-valued Mercer kernel with RKHS $\mathcal{V}$. In this case, the RKHS $\boldsymbol{\mathcal{V}}$ can be identified as $\boldsymbol{\mathcal{V}} = \{v : \Omega \to \mathbb{R}^d \mid v_i \in \mathcal{V}, \quad i = 1, \dots, d\}$ where we used v_i to denote the i-th component of v, i.e., $v(x) = (v_1(x), \dots, v_d(x))$. Furthermore, we have that $\|v\|^2_{\boldsymbol{\mathcal{V}}} = \sum_{i=1}^d \|v_i\|^2_{\mathcal{V}}$ and $\langle v, v' \rangle_{\boldsymbol{\mathcal{Q}}} = \sum_{i=1}^d \langle v_i, v_i' \rangle_{\mathcal{V}}$, $\forall v, v' \in \boldsymbol{\mathcal{V}}$.

2.3.2 Error analysis

We dedicate this section to the proof of Main Theorem 1. We present a sequence of propositions and lemmas that are of independent interest and from which the proof of Main Theorem 1 follows as a corollary.

Let $\Omega \subset \mathbb{R}^d$ be a bounded open set and define the space

$$\mathbb{L}_0(\Omega; \mathbb{R}^d) := \left\{ f : \Omega \to \mathbb{R}^d \;\middle|\; \|f\|_{\mathbb{L}(\Omega;\mathbb{R}^d)} < +\infty \quad \text{and} \quad f(x) = 0 \quad \forall x \in \Omega^c \right\},$$

where $\|f\|_{\mathbb{L}(\Omega;\mathbb{R}^d)} := \sup_{x \in \Omega} |f(x)| + \sup_{x,y \in \Omega} \frac{|f(x)-f(y)|}{|x-y|}$, that is, $\mathbb{R}^d$ valued functions on Ω that are bounded and Lipschitz, and vanish outside of Ω. Here $|\cdot|$ denotes the Euclidean norm on $\mathbb{R}^d$. Further define $\mathbb{V} := L^1([0,1]; \mathbb{L}_0(\Omega; \mathbb{R}^d))$, the space of Bochner integrable vector fields on the (time) interval $[0,1]$, taking values in $\mathbb{L}_0(\Omega; \mathbb{R}^d)$, with the norm defined as $\|f\|_{\mathbb{V}} = \int_0^1 \sup_{x \in \Omega} |v(t,x)| dt + \int_0^1 \sup_{x,y \in \Omega} \frac{|v(t,x)-v(t,y)|}{|x-y|} dt$. Let us now define $\Gamma := [0,1] \times \Omega$ and consider a matrix-valued Mercer kernel $\boldsymbol{Q} : \Gamma \times \Gamma \to \mathbb{R}^{d \times d}$ with RKHS $\boldsymbol{\mathcal{Q}}$. The following lemma follows directly from classic results from ODE theory; see for example [234, Thms. C.3 and 8.7]:

Lemma 2.1. *Suppose $\boldsymbol{\mathcal{Q}} \subseteq \mathbb{V}$ and consider an ODE of the form $\phi_t = v(t, \phi)$, with $\phi(0, x) = x$ for $v \in \boldsymbol{\mathcal{Q}}$. Then (i) for all $x \in \Omega$ there exists a unique solution $\phi(t, x)$ of the ODE on $[0,1]$ and (ii) the flow map associated with the ODE is at all times continuous, invertible, differentiable and has a continuous, differentiable inverse (i.e., a diffeomorphism of Ω).*

Given a set of inducing points $S := \{s_1, \dots, s_M\} \subset \Gamma$ and a scalar $r > 0$ define the sets $\boldsymbol{\mathcal{Q}}_r^S \subset \boldsymbol{\mathcal{Q}}_r \subset \boldsymbol{\mathcal{Q}}$ as

$$\boldsymbol{\mathcal{Q}}_r := \left\{ v \in \boldsymbol{\mathcal{Q}} \;\middle|\; \|v\|^2_{\boldsymbol{\mathcal{Q}}} \le r^2 \right\}, \qquad \boldsymbol{\mathcal{Q}}_r^S := \left\{ v = \sum_{j=1}^M c_j \boldsymbol{Q}(s_j, \cdot) \;\middle|\; C^T \boldsymbol{Q}(S,S) C \le r^2 \right\}.$$

Let us now recall the setting of Problem Eq (2.6) by considering a Mercer kernel $K : \Omega \times \Omega \to \mathbb{R}$ and reference and target measures $\eta, \nu \in \mathbb{P}(\Omega)$ with their N-sample empirical approximations η^N, ν^N.

We then define

$$v_r := \begin{cases} \underset{v\in\mathbb{V}}{\arg\min} & \mathrm{MMD}_K(\phi(1,\cdot)\#\eta,\nu), \\ \text{s. t.} & \phi_t = v(t,\phi), \quad \phi(0,x) = x\,\forall x\in\Omega, \quad v\in\boldsymbol{\mathcal{Q}}_r, \end{cases} \tag{2.8a}$$

$$v_r^S := \begin{cases} \underset{v\in\mathbb{V}}{\arg\min} & \mathrm{MMD}_K(\phi(1,\cdot)\#\eta,\nu), \\ \text{s. t.} & \phi_t = v(t,\phi), \quad \phi(0,x) = x\,\forall x\in\Omega, \quad v\in\boldsymbol{\mathcal{Q}}_r^S, \end{cases} \tag{2.8b}$$

$$v_r^{S,N} := \begin{cases} \underset{v\in\mathbb{V}}{\arg\min} & \mathrm{MMD}_K(\phi(1,\cdot)\#\eta^N,\nu^N), \\ \text{s. t.} & \phi_t = v(t,\phi), \quad \phi(0,x) = x\,\forall x\in\Omega, \quad v\in\boldsymbol{\mathcal{Q}}_r^S. \end{cases} \tag{2.8c}$$

noting that $v_r^{S,N}$ is precisely the velocity field that solves Eq (2.6). By Lemma 2.1 above, all of these problems are feasible since the ODEs for ϕ are well-defined and have unique solutions for the prescribed class of velocity fields.

Remark 2.2. *We note that the problems in Eq (2.8) may have multiple minimizers due to the fact that the loss functions are not convex even if we were to choose a minimum $\boldsymbol{\mathcal{Q}}$ norm solution. For this reason, we simply take v_r, v_r^S, and $v_r^{S,N}$ to be any minimizer of these problems in the event of multiple minimizers moving forward.*

We now wish to show that the problems in Eq (2.8) attain their minimizers under some mild assumptions and so the velocity fields $v_r, v_r^S, v_r^{S,N}$ are well-defined. To this end, for a fixed $v\in\mathbb{V}$ define the transport map

$$T(x;v) := \phi(1,x), \quad \text{s. t.} \quad \phi_t = v(t,\phi), \quad \phi(0,x) = x.$$

We then have the following lemma as a consequence of classic perturbation results for ODEs with respect to the velocity field v (see [142, Thm. 4.7]), stating that $T(x;v)$ is continuous in both of its arguments:

Lemma 2.3. *Suppose $v, v'\in\mathbb{V}$ and let $L_v := \sup_{s,s'\in\Gamma}\frac{|v(s)-v(s')|}{|s-s'|}$. Then it holds that*

$$|T(x;v) - T(x';v')| \le \exp(L_v)|x-x'| + \frac{(\exp(L_v)-1)}{L_v}\|v-v'\|_\infty,$$

where we introduced the notation $\|v-v'\|_\infty := \sup_{s\in\Gamma}|v-v'|$.

We also recall the following stability result for MMD [14, Thm. 3.2]:

Lemma 2.4. *Suppose $K : \Omega \times \Omega \to \mathbb{R}$ is a stationary Mercer kernel such that*

$$\sup_{x,x' \in \Omega} \frac{\|K(x,\cdot) - K(x',\cdot)\|_{\mathcal{K}}}{|x - x'|} \leq L_K \tag{2.9}$$

with a constant $L_K > 0$. For a measure $\eta \in \mathbb{P}(\Omega)$ and exponent $p \in [1, +\infty]$ consider the space $L^\infty_\eta(\Omega;\Omega) := \{T : \Omega \to \Omega \mid |||T\|_{L^\infty_\eta(\Omega;\Omega)} < +\infty\}$ [3] with norm $\|T\|_{L^\infty_\eta(\Omega;\Omega)} := \operatorname{ess\,sup}_{x \in supp\ (\eta)} |T(x)|$. Then it holds, for any pair of maps $T, T' \in L^\infty_\eta(\Omega;\Omega)$, that

$$\mathrm{MMD}_K(T\#\eta, T'\#\eta) \leq L_K \|T - T'\|_{L^\infty_\eta(\Omega;\Omega)}.$$

With the above lemmas at hand, we can now prove the continuous dependence of the MMD loss function in Eq (2.8) on the underlying velocity fields.

Proposition 2.5. *Suppose $v, v' \in \mathbb{V}$ and K is a Mercer kernel satisfying Eq (2.9) with a constant $L_K > 0$. Then for pairs of measures $\eta, \nu \in \mathbb{P}(\Omega)$ it holds that*

$$|\mathrm{MMD}_K(T(\cdot;v)\#\eta, \nu) - \mathrm{MMD}_K(T(\cdot;v')\#\eta, \nu)| \leq \frac{L_K(\exp(L_v) - 1)}{L_v} \|v - v'\|_\infty.$$

Proof. Since MMD_K satisfies the triangle inequality and is symmetric, we have that

$$|\mathrm{MMD}_K(T(\cdot;v)\#\eta, \nu) - \mathrm{MMD}_K(T(\cdot;v')\#\eta, \nu)| \leq \mathrm{MMD}_K(T(\cdot;v)\#\eta, T(\cdot;v')\#\eta).$$

Then applying Lemma 2.4 followed by Lemma 2.3 with $x = x'$ gives the result. □

As a result of the above proposition, we can now establish that under some regularity assumptions, the problems in Eq (2.8) have feasible minimizers.

Proposition 2.6. *Suppose $K : \Omega \times \Omega \to \mathbb{R}$ is a stationary Mercer kernel satisfying Eq (2.9), let $\eta, \nu \in \mathbb{P}_K(\Omega)$, and assume $\boldsymbol{\mathcal{Q}}$ is compactly embedded in $\mathbb{V}$, i.e., $\boldsymbol{\mathcal{Q}}$ is a compact subset of $\mathbb{V}$ and there exists a constant $C_{\boldsymbol{\mathcal{Q}}} > 0$ so that $\|v\|_{\mathbb{V}} \leq C_{\boldsymbol{\mathcal{Q}}} \|v\|_{\boldsymbol{\mathcal{Q}}}$ for all $v \in \boldsymbol{\mathcal{Q}}$. Then the minimization problems in Eq (2.8) have feasible minimizers.*

[3] This result can also be stated for maps in $L^p_\eta(\Omega;\Omega)$, but L^∞ is natural here since our maps are continuous.

Proof. Since $\eta, \nu \in \mathbb{P}_K(\Omega)$ then $\mathrm{MMD}_K(\eta, \nu) < +\infty$ but $\eta = T(\cdot; 0)\#\eta$, i.e., the velocity field arising from the zero vector field. Since 0 is an element of both spaces $\boldsymbol{\mathcal{Q}}_r$ and $\boldsymbol{\mathcal{Q}}_r^S$ then all three problems are feasible. The same argument also applies to η^N, ν^N, in fact, in this case it always holds that $\eta^N, \nu^N \in \mathbb{P}_K(\Omega)$. Then Proposition 2.5 and the assumption that $\boldsymbol{\mathcal{Q}}_r$ (and similarly $\boldsymbol{\mathcal{Q}}_r^S$) are compact subsets of $\mathbb{V}$ give the existence of minimizers in Eq (2.8); see for example [186, Thm. 1.9]. □

We now turn our attention to controlling the error of the transport map arising from the velocity field $v_r^{S,N}$, which is an object that we can compute in practice. This velocity field is random due to its dependence on the empirical measures η^N, ν^N. Therefore, we aim to obtain a high-probability bound on the quantity $\mathrm{MMD}_K(T(\cdot; v_r^{S,N})\#\eta, \nu)$. To achieve such a bound, we rely on [37, Thm. 1], which gives a generalization bound for minimum MMD generative models. We state that result as a lemma below for convenience:

Lemma 2.7. *Let $K : \Omega \times \Omega \to \mathbb{R}$ be a bounded Mercer kernel and let $\mu \in \mathbb{P}_K(\Omega)$. Consider a generative model (a parametric probability measure) $\mu_\theta \in \mathbb{P}_K(\Omega)$ parameterized by $\theta \in \Theta$, taken to be an arbitrary Banach space. Suppose it holds that*

(i) For every $\mu \in \mathbb{P}_K(\Omega)$, there exists $C > 0$ such that the set $\{\theta \in \Theta \mid \mathrm{MMD}_K(\mu_\theta, \mu) \leq \inf_{\theta\in\Theta} \mathrm{MMD}_K(\mu_\theta, \mu) + C\}$ is bounded.

(ii) For every $N > 0$ and $\mu \in \mathbb{P}_K(\Omega)$, there exists $C_N > 0$ such that the set $\{\theta \in \Theta \mid \mathrm{MMD}_K(\mu_\theta^N, \mu) \leq \inf_{\theta\in\Theta} \mathrm{MMD}_K(\mu_\theta, \mu) + C_N\}$ is almost surely bounded, where μ_θ^N is an empirical approximation to μ_θ.

Define $\theta^N := \arg\min_{\theta\in\Theta} \mathrm{MMD}_K(\mu_\theta, \mu^N)$. Then, with probability at least $1 - \delta$ we have

$$\mathrm{MMD}_K(\mu_{\theta^N}, \mu) \leq \inf_{\theta\in\Theta} \mathrm{MMD}_K(\mu_\theta, \mu) + 2\sqrt{\frac{2}{N}\sup_{x\in\Omega} K(x,x)}\left(2 + \sqrt{\log\left(\frac{1}{\delta}\right)}\right).$$

Remark 2.8. *Note that in the original article [37], this theorem is stated for Θ being a finite-dimensional Euclidean space, however an inspection of the proof reveals that this assumption is not needed and therefore, we state the result assuming Θ is a Banach space.*

We apply Lemma 2.7 with $\mu \leftarrow \nu$, and $\mu_\theta \leftarrow T(\cdot; v)\#\eta$ for $v \in \boldsymbol{\mathcal{Q}}_r^S$, i.e., our generative models are parameterized by the velocity fields v and so $\Theta \leftarrow \boldsymbol{\mathcal{Q}}_r^S$.

Proposition 2.9. *Suppose $K : \Omega \times \Omega \to \mathbb{R}$ is a Mercer kernel that satisfies Lemma 2.4, $\eta, \nu \in \mathbb{P}_K(\Omega)$, and $\boldsymbol{\mathcal{Q}}$ is compactly embedded in $\mathbb{V}$. Then with probability at least $1-\delta$, for $\delta > 0$, it holds that*

$$\mathrm{MMD}_K(T(\cdot; v_r^{S,N})\#\eta, \nu) \leq \mathrm{MMD}_K(T(\cdot; v_r^S)\#\eta, \nu) + 2\sqrt{\frac{2}{N}\sup_{x\in\Omega} K(x,x)}\left(2 + \sqrt{\log\left(\frac{1}{\delta}\right)}\right). \quad (2.10)$$

Proof. Since Ω is assumed to be bounded then the Lipschitz assumption on κ implies that K is bounded as well. Further, since $\boldsymbol{\mathcal{Q}}_r$ is readily bounded, then conditions (i, ii) of Lemma 2.7 are satisfied for our setup. Applying that lemma together with the fact that by Proposition 2.6 we have $\inf_{v\in\boldsymbol{\mathcal{Q}}_r^S} \mathrm{MMD}_K(T(\cdot; v)\#\eta, \nu) = \mathrm{MMD}_K(T(\cdot; v_r^S)\#\eta, \nu)$ gives the result. □

We now turn our attention to the first term in Eq (2.10), which will reflect the approximation power of the class $\boldsymbol{\mathcal{Q}}_r^S$. We will bound this term under stronger smoothness assumptions on the space $\boldsymbol{\mathcal{Q}}$ and, in particular, focus on the case of diagonal kernels.

Proposition 2.10. *Suppose $K : \Omega \times \Omega \to \mathbb{R}$ is a Mercer kernel that satisfies Lemma 2.4 and let $\eta, \nu \in \mathbb{P}_K(\Omega)$. Furthermore, let $V : \Gamma \times \Gamma \to \mathbb{R}$ be another Mercer kernel such that $V \in C^{2k}(\Gamma \times \Gamma)$ for some $k \geq 1$ with $\mathcal{V}$ as its RKHS. Let $\boldsymbol{\mathcal{Q}}(s, s') = V(s, s')I$ be the corresponding diagonal, matrix-valued kernel defined from V and take $\boldsymbol{\mathcal{Q}}$ to be its RKHS which is assumed to be compactly embedded in $\mathbb{V}$. Suppose $S = \{s_1, \ldots, s_M\} \subset \Gamma$ is a collection of distinct points with fill distance $h_S := \sup_{s\in\Gamma} \inf_{s'\in S} \|s - s'\|_2$. Then there exists $h_0 > 0$ such that for $h_S \leq h_0$ it holds that*

$$\mathrm{MMD}_K(T(\cdot; v_r^S)\#\eta, \nu) \leq CL_K(\exp(C_{\boldsymbol{\mathcal{Q}}} r) - 1)h_S^k + \mathrm{MMD}_K(T(\cdot; v_r)\#\eta, \nu)$$

where $C_{\boldsymbol{\mathcal{Q}}} > 0$ is the embedding constant of $\boldsymbol{\mathcal{Q}}$ and $C > 0$ is independent of h_S, r, v_r.

Proof. Under the hypothesis of the theorem on K and κ we have that $\boldsymbol{\mathcal{Q}} \subset \mathbb{V}$. Now consider the velocity field

$$\widehat{v}_r^S := \operatorname*{arg\,min}_{v\in\mathbb{V}} \|v\|_{\boldsymbol{\mathcal{Q}}} \quad \text{s. t.} \quad v(s_j) = v_r(s_j), \quad j = 1, \ldots, M,$$

which is simply the interpolant of v_r on S in $\boldsymbol{\mathcal{Q}}$. Note that $\widehat{v}_r^S \in \boldsymbol{\mathcal{Q}}_r^S$ since $\|\widehat{v}_r^S\|_{\boldsymbol{\mathcal{Q}}} \leq \|v_r\|_{\boldsymbol{\mathcal{Q}}}$. Then using the optimality of v_r^S (recall Eq (2.8)) along with the triangle inequality for MMD_K we obtain

$$\mathrm{MMD}_K(T(\cdot; v_r^S)\#\eta, \nu) \leq \mathrm{MMD}_K(T(\cdot; \widehat{v}_r^S)\#\eta, T(\cdot; v_r)\#\eta) + \mathrm{MMD}_K(T(\cdot; v_r)\#\eta, \nu). \quad (2.11)$$

Let us now focus on the first term on the right-hand side. Applying Lemma 2.4 and Lemma 2.3 in that order yields

$$\mathrm{MMD}_K(T(\cdot;\widehat{v}_r^S)\#\eta, T(\cdot;v_r)\#\eta) \leq L_\kappa \frac{\exp(L_{v_r})-1}{L_{v_r}} \|\widehat{v}_r^S - v_r\|_\infty.$$

Since we assumed $\boldsymbol{\mathcal{Q}}$ is compactly embedded in $\mathbb{V}$ then $L_{v_r} \leq \|v_r\|_{\mathbb{V}} \leq C_{\boldsymbol{\mathcal{Q}}}\|v_r\|_{\boldsymbol{\mathcal{Q}}} \leq C_{\boldsymbol{\mathcal{Q}}} r$. Observing that $\frac{1}{t}(\exp(t)-1)$ is monotone increasing for $t>0$ we obtain the bound

$$\mathrm{MMD}_K(T(\cdot;\widehat{v}_r^S)\#\eta, T(\cdot;v_r)\#\eta) \leq L_K \frac{\exp(C_{\boldsymbol{\mathcal{Q}}} r)-1}{C_{\boldsymbol{\mathcal{Q}}} r} \|\widehat{v}_r^S - v_r\|_\infty. \tag{2.12}$$

It remains for us to bound the approximation error $\|\widehat{v}_r^S - v_r\|_\infty$. Recall that by definition

$$\|\widehat{v}_r^S - v_r\|_\infty = \sup_{s\in\Gamma} |\widehat{v}_r^S(s) - v_r(s)| = \sup_{s\in\Gamma} \left(\sum_{j=1}^d |(\widehat{v}_{r,j}^S(s) - v_{r,j}(s)|^2 \right)^{1/2}. \tag{2.13}$$

On the other hand, since we assumed $\boldsymbol{\mathcal{Q}}$ is a diagonal kernel then $\widehat{v}_{r,j}^S$ is precisely the $\mathcal{V}$ interpolant of $v_{r,j}$ for $j=1,\ldots,d$. Then by [228, Thm. 11.13] we have that $\exists h_0 > 0$ such that for $h_S \leq h_0$ we have

$$\|\widehat{v}_{r,j}^S - v_{r,j}\|_\infty \leq C h_S^k \|v_{r,j}\|_{\mathcal{V}},$$

for a constant $C>0$ that is independent of h_S and $v_{r,j}$. Substituting this bound in Eq (2.13) we obtain the error bound

$$\|\widehat{v}_r^S - v_r\|_\infty \leq C h_S^k \left(\sum_{j=1}^d \|v_{r,j}\|_{\mathcal{V}}^2 \right)^{1/2} = C h_S^k \|v_r\|_{\boldsymbol{\mathcal{Q}}} \leq C h_S^k r$$

Finally, substituting this result in (2.12) and plugging that back into (2.11) yields the result. □

Combining Propositions 2.9 and 2.10 we can finally state our main theoretical result, which serves as the precise version of Main Theorem 1:

Corollary 2.11. *Suppose Propositions 2.9 and 2.10 are satisfied and that there exists a vector field $v^\dagger \in \mathbb{V}$ such that $T(\cdot;v^\dagger)\#\eta = \nu$. Then with probability $1-\delta$, for $\delta \in (0,1)$, it holds that*

$$\begin{aligned}\mathrm{MMD}_K(T(\cdot;v_r^{S,N})\#\eta, \nu) \leq C \Bigg[&(\exp(C_{\boldsymbol{\mathcal{Q}}} r)-1) h_S^\ell + \frac{\exp(L_{v^\dagger})-1}{L_{v^\dagger}} \inf_{v\in\boldsymbol{\mathcal{Q}}_r} \|v - v^\dagger\|_\infty \\ &+ \sqrt{\frac{1}{N}} \left(2 + \sqrt{\log\left(\frac{1}{\delta}\right)} \right) \Bigg],\end{aligned} \tag{2.14}$$

where $C>0$ is independent of S, r, N and $v^\dagger$.

Proof. Applying Proposition 2.9 and Proposition 2.10 and combining the independent constants into $C > 0$ yields the same bound as Eq (2.14) with the bias term $\inf_{v\in\boldsymbol{\mathcal{Q}}_r} \|v - v^\dagger\|_\infty$ replaced by $\mathrm{MMD}_K(T(\cdot; v_r)\#\eta, \nu)$. Let $v_r^\dagger = \min_{v\in\boldsymbol{\mathcal{Q}}_r} \|v - v^\dagger\|_\infty$ and observe that the optimality of v_r implies that

$$\mathrm{MMD}_K(T(\cdot; v_r)\#\eta, \nu) \leq \mathrm{MMD}_K(T(\cdot; v_r^\dagger)\#\eta, \nu) = \mathrm{MMD}_K(T(\cdot; v_r^\dagger)\#\eta, T(\cdot; v^\dagger)\#\eta).$$

Applying Lemma 2.4 and Lemma 2.3 further gives the sequence of bounds

$$\begin{aligned}\mathrm{MMD}_K(T(\cdot; v_r^\dagger)\#\eta, T(\cdot; v^\dagger)) &\leq L_K \|T(\cdot; v_r^\dagger) - T(\cdot; v^\dagger)\|_{L^\infty_\eta(\Omega;\Omega)} \\ &\leq L_K \frac{(\exp(L_{v^\dagger}) - 1}{L_{v^\dagger}} \|v_r^\dagger - v^\dagger\|_\infty.\end{aligned}$$

which yields the desired result. □

2.3.3 Unconstrained minimization with a regularization term

So far, our theoretical analysis has been focused on the error analysis of problem Eq (2.8c) (which also coincides with Eq (2.6)), however our numerical algorithms are based on the unconstrained relaxation Eq (2.5), which we recall as

$$v_\lambda^{S,N} := \begin{cases} \underset{v\in\mathbb{V}}{\arg\min} & \mathrm{MMD}_K(\phi(1,\cdot)\#\eta^N, \nu^N) + \lambda \sum_{\ell=1}^d \|v_\ell\|_{\mathcal{V}}^2 \\ \text{s. t.} & \phi_t = v(t,\phi), \quad \phi(0,x) = x \quad \forall x \in \Omega, \\ & v(s) = (v_1(s), \ldots, v_d(s)) \quad v_\ell = c_\ell^T V(S,\cdot). \end{cases} \tag{2.15}$$

Note our abuse of notation here with $v_r^{S,N}$ denoting the solution to Eq (2.8c) while $v_\lambda^{S,N}$ denotes its unconstrained relaxation in Eq (2.15). Since $v_\lambda^{S,N}$ is optimal we immediately obtain the bound

$$\mathrm{MMD}_K(T(\cdot; v_\lambda^{S,N})\#\eta^N, \nu^N) \leq \mathrm{MMD}_K(T(\cdot; v_r^{S,N}\#\eta^N, \nu^N) + \lambda \max\left\{0, \sum_{\ell=1}^d \|v_{r,\ell}^{S,N}\|_{\mathcal{V}}^2 - \|v_{\lambda,\ell}^{S,N}\|_{\mathcal{V}}^2\right\}.$$

This suggests that the bound in Eq (2.14) can be extended to $v_\lambda^{S,N}$ up to a (possibly non-zero) bias term concerning the choice of λ and r.

2.4 Numerical Implementation

In this section, we collect details around the numerical implementation of problem Eq (2.5) and discuss our strategies in preparation for the benchmark examples in Section 2.5.

2.4.1 Summary of the algorithm

Noting that Eq (2.5) can readily be implemented by discretizing the ODE, we focus our attention here on the choice of kernels and tuning of hyper-parameters.

The choice of the kernel V We will work with kernels $V(s, s')$ that are of product form in space and time, that is,

$$V(s, s') \equiv V((t, x), (t', x')) = W(t, t')U(x, x')$$

for Mercer kernels $W : [0, 1] \times [0, 1] \to \mathbb{R}$ and $U : \Omega \times \Omega \to \mathbb{R}$ with RKHS spaces $\mathcal{W}$ and $\mathcal{U}$, respectively. Choosing $\mathcal{W} = H^1([0, 1])$ yields the particularly useful form

$$\|v_\ell\|_{\mathcal{V}}^2 = \lambda_1 \int_0^1 \|v_\ell(t, \cdot)\|_{\mathcal{U}}^2 \, \mathrm{d}t + \lambda_2 \int_0^1 \|\dot{v}_\ell(t, \cdot)\|_{\mathcal{U}}^2 \, \mathrm{d}t, \qquad \ell = 1, \ldots, d, \tag{2.16}$$

where $\dot{v}_\ell$ denotes the time derivative of v_ℓ and $\lambda_1, \lambda_2 > 0$ are constants. We note that the second term is non-standard in the context of diffeomorphic matching [234, Ch. 10] or OT [154] (see also [220, pp. 159]). In our numerical experiments in Section 2.5 we will present ablation studies that demonstrate the effect of the second term leading to smoother flow maps; see Figure 2.3. We will also consider settings where $\mathcal{W}$ consists of constant functions (i.e., $\dot{v}_\ell$ is maximally penalized) in which case the velocity field is constant in time, leading to an autonomous ODE in Eq (2.5). We will refer to this setting as *Autonomous KODE*. For the choice of the kernel U we often use standard choices such as the Gaussian/squared exponential kernel or the Laplace kernel

$$U_{\text{Gaussian}}(x, x') = \exp\left(-\frac{|x - x'|^2}{2\gamma_U^2}\right) \qquad U_{\text{Laplace}}(x, x') = \exp\left(-\frac{|x - x'|}{\gamma_U}\right) \tag{2.17}$$

where in both cases $\gamma > 0$ denotes a lengthscale parameter.

The choice of the inducing points S Since the kernel V is of product form, it is natural for us to also choose S to be of a similar structure. To this end, we choose a set of spatial inducing points $X = \{x_1, \ldots, x_J\} \subset \Omega$ using k-means on the combined samples from both reference and target measures [242]. We write $v_\ell(t, x) = c_j(t)^T U(X, x)$ for a set of coefficient vector fields $c_j : [0, 1] \to \mathbb{R}^J$. This leads to a spatial discretization of Eq (2.16) in the following form:

$$\|v_\ell\|_{\mathcal{V}}^2 \approx \lambda_1 \int_0^1 c_\ell(t)^T U(X, X) c_\ell(t) \, \mathrm{d}t + \lambda_2 \int_0^1 \dot{c}_\ell(t)^T U(X, X) \dot{c}_\ell(t) \, \mathrm{d}t, \qquad \ell = 1, \ldots, d.$$

The coefficient functions $c_\ell(t) : [0,1] \to \mathbb{R}^J$ can be viewed as the trajectories of a a system of coupled ODEs. We further discretize the above integrals using the mid-point rule with intervals of size Δt (chosen so that $1/\Delta t$ is an integer), to obtain the discrete RKHS penalty, which is implemented in our code

$$\|v_\ell\|_{\mathcal{V}}^2 \approx \Delta t \sum_{k=1}^{1/\Delta t} \left(\lambda_1 c_{\ell,k}^T U(X,X) c_{\ell,k} + \lambda_2 \dot{c}_{\ell,k}^T U(X,X) \dot{c}_{\ell,k}\right), \qquad \ell = 1, \ldots, d.$$

The time derivatives $\dot{c}_{\ell,j}$ are further computed using standard finite-difference formulae such as centered differences in the interior of the unit interval and one-sided formulae at the boundaries. The aforementioned discretization of the RKHS penalty arises from choosing a set of space-time inducing points S on a lattice obtained by tensorizing X with a uniform grid of points in time. More precisely, let $t_k = k\Delta t$ for $k = 0, \ldots, 1/\Delta t$. Then, defining $s_{j,k} = (x_j, t_k)$ we obtain the set of inducing points $S = \{s_{j,k}\}_{j=1,k=1}^{j=J,k=1/\Delta t}$ that is consistent with our error analysis in Section 2.3.

The choice of the kernel K In all of our experiments, we take the MMD kernel K to belong to the radial family, and in particular, we take K to be the Laplace kernel as in Eq (2.17). We note that the choice of the Gaussian kernel or the well-known inverse quadratic kernel is also possible, although we found that the differences were minimal. It is important to note that in our framework, the choice of the kernels K and U are not related, and the parameters of these kernels can be tuned independently of each other.

Tuning hyper-parameters Our model contains a number of hyper-parameters such as the regularization parameters λ_1 and λ_2, the lengthscales γ_U and γ_K for the kernels U, K respectively, and the step size Δt. We utilized standard cross-validation techniques and the median heuristic for choosing our kernel lengthscales [95]; details can be found in our repository [4]. We selected λ_1, λ_2 through grid search within the range of $[10^{-2}, 10^{-10}]$, optimizing for the best performance on the validation set.

[4] `https://github.com/TADSGroup/KernelODETransport`

2.5 Experiments

Below, we collect the results of our numerical experiments on a collection of benchmark problems. In all of our examples, we take the reference measure η to be a standard Gaussian distribution.

2.5.1 Overview of the experiments

First, we evaluated the performance of KODE for sampling two-dimensional measures that are standard benchmarks for generative modeling tasks [94]. In these seven examples, the target measures ν are concentrated on complex shapes like a pinwheel or a checkerboard. Our data sets for these experiments consist of 25,000 samples drawn from ν that are split into training, validation, and test samples. We used 5000 training samples for all of our examples except for the checkerboard data set, for which 10000 samples were used (the latter is the most challenging of the 2D benchmarks). The 2D benchmark results are collected in Section 2.5.2.

Next, we applied KODE to higher-dimensional benchmark data sets: `POWER`, `GAS`, `HEPMASS`, and `MINIBOONE` from the University of California Irvine (UCI) machine learning data repository and the Berkeley Segmentation Dataset (`BSDS300`) from UC Berkeley Computer Vision group; all of these are commonly used benchmarks in the normalizing flow literature [123, 164]. These data sets range from 6 to 63 dimensions and have distributions with varying levels of complexity. All of these data sets were implemented using their off-the-bench training and testing splits, and the results are collected in Section 2.5.3

We also performed two experiments in addition to the standard benchmarks above. In Section 2.5.3 we present an example of image generation for the MNIST data set by augmenting KODE with neural net features that are lightly trained. In Section 2.5.4 we present use a small modification of KODE to obtain a triangular map that is capable of likelihood-free inference and use it to infer the parameters of a Lotka-Volterra ODE.

Comparison metrics We compared the performance of KODE with the OT-Flow algorithm of [154], which is also a method based on the dynamic formulation of transport although it utilizes neural nets to parameterize the velocity fields. To be fair, in comparison of the time and complexity of the models, we trained OT-Flow using code from the original article on the same hardware that

KODE was trained on.

In order to compare the quality of the produced samples for both methods, we used the MMD metric alongside the Negative Log Likelihood (NLL). In the context of normalizing flows, NLL measures how well a model represents the target distribution ν with density ρ_0. Specifically, an invertible mapping T is learned between the target distribution ν and a standard normal distribution η with density ρ_1, by minimizing the NLL of the target data. Using the change of variables formula, the log-likelihood of a data point x is expressed as:

$$\log \rho_0(x) = \log \rho_1(T(x)) + \log |\det \nabla T(x)|. \tag{2.18}$$

Once the mapping T is learned, new samples from ν can be generated by applying T^{-1} to samples from η. For a more comprehensive review of normalizing flows, we refer the reader to [123].

In the experiments, we made sure the kernel for the MMD metric is different from the one used in the training of KODE to be fair to OT-Flow. Indeed, following [154] we used the Gaussian kernel with a unit lengthscale for this purpose while KODE was trained using the Laplace kernel. Finally, in all of our experiments we report the normalized MMD values which denotes the MMD between the generated samples and the test data set normalized by the MMD between reference samples and the test data.

2.5.2 2D benchmarks

We start by comparing autonomous and non-autonomous KODE with OT-Flow on benchmark examples in two dimensions. Table 2.1 compares these two versions of KODE with OT-Flow in terms of the number of parameters, training wall-clock time, normalized MMD, and NLL of the generated samples. Clearly, the autonomous KODE has the lowest number of parameters since it does not require time discretization of the coefficient functions $c_\ell(t)$. We observed that KODE performed on par with OT-Flow on all benchmarks. The case of the *checkerboard* measure is particularly interesting since KODE appears to outperform OT-Flow in terms of normalized MMD by a large margin despite having comparable negative log-likelihood. In all examples, we tried to match the number of degrees of freedom in non-autonomous KODE and OT-Flow, but we observed that the training wall-clock time for KODE was generally significantly shorter than OT-Flow. The autonomous KODE algorithm is much faster to train as expected, although this efficiency comes

at the cost of test performance except for the *circles* data set where autonomous KODE appears to achieve the best normalized MMD performance despite being much simpler.

Figure 2.1 shows generated samples from non-autonomous KODE. The top row shows the target measure ν, the middle row shows the KODE samples, and the bottom row shows the samples generated by pulling the target samples to the Gaussian reference by running the KODE model backward in time after training. In all examples, there is a good match between the target measure and the pushforward measure and the pullback (i.e., reverse transport) recovers the Gaussian reference measure with good quality.

We further tested the effect of the time derivative penalties in our regularization term Eq (2.16) for the velocity fields. In Figure 2.3 we show the trajectories of a set of samples from the reference to the target generated by KODE with and without penalizing the time derivatives, i.e., $\lambda_2 \neq 0$ or $\lambda_2 = 0$. We clearly observe that the sample trajectories are smoother in the latter case, implying that the resulting velocity fields are smoother and can be simulated more efficiently using adaptive ODE solvers.

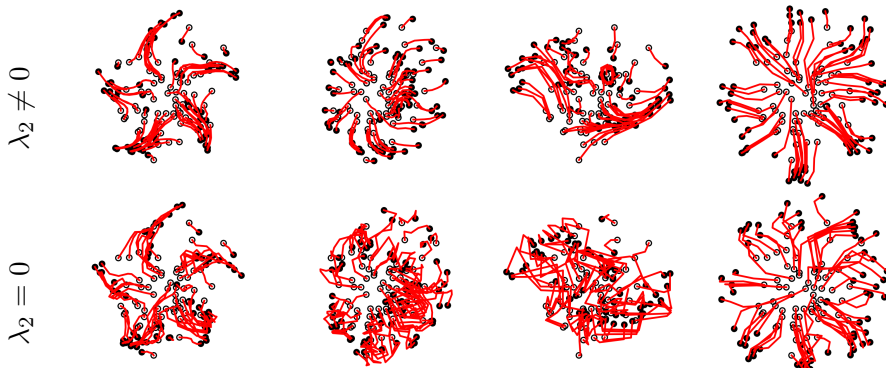

Figure 2.3: **KODE sample trajectories for 2D benchmarks with different choices of λ_2 in Eq (2.16)**: (Top row) using $\lambda_2 \neq 0$ so the time derivatives of the coefficients are penalized; (Bottom row) using $\lambda_2 = 0$.

2.5.3 Higher-dimensional benchmarks

Next, we compared KODE with OT-Flow on high-dimensional data sets. Table 2.2 shows our detailed quantitative comparisons akin to the 2D benchmarks. We found that non-autonomous

Table 2.1: **Summary of numerical results on 2D benchmark examples**: we report the normalized MMD, negative log-likelihood (NLL), number of trainable model parameters, and the total training time. We compare autonomous and non-autonomous KODE models with the OT-Flow model.

Dataset	Model	# Parameters	Training time (h)	Normalized MMD	NLL
Pinwheel	KODE (autonomous)	800	0.13	4.0e-3	2.98
	KODE (non-autonomous)	1000	0.13	3.4e-3	2.45
	OT-FLOW	**1229**	**0.25**	**1.8e-3**	**2.34**
2spirals	KODE (auto)	400	0.06	1.0e-2	4.04
	KODE (non-auto)	1000	0.13	7.1e-3	2.91
	OT-FLOW	**1229**	**0.22**	**6.2e-3**	**2.67**
moons	KODE (auto)	200	0.06	9.8e-3	2.61
	KODE (non-auto)	**900**	**0.12**	**4.1e-3**	2.48
	OT-FLOW	1229	0.26	5.7e-3	**2.44**
8gaussians	KODE (auto)	800	0.13	1.1e-3	3.00
	KODE (non-auto)	900	0.18	9.5e-4	2.91
	OT-FLOW	**1229**	**0.25**	**4.0e-4**	**2.85**
circles	**KODE (auto)**	**800**	**0.08**	**3.4e-3**	4.13
	KODE (non-auto)	1000	0.14	4.5e-3	3.34
	OT-FLOW	1229	0.23	7.2e-3	**3.28**
swissroll	KODE (auto)	800	0.08	5.3e-3	3.65
	KODE (non-auto)	1000	0.14	4.6e-3	2.77
	OT-FLOW	**1229**	**0.25**	**3.0e-3**	**2.69**
checkerboard	KODE (auto)	1000	0.13	1.7e-3	3.87
	KODE (non-auto)	**1200**	**0.27**	**7.2e-4**	3.78
	OT-FLOW	1229	1.65	1.7e-3	**3.53**

KODE achieves competitive performance in three out of five examples for the MMD metric. In particular, for the `POWER` data set, KODE significantly outperformed OT-Flow with a comparable training time while using a third of the parameters. For the `GAS` and `BSDS300` data sets, KODE was on par with OT-FLOW albeit using fewer parameters and faster training time in some cases. For

Table 2.2: **Summary of numerical results on high-dimensional benchmark examples**: we report the normalized MMD, negative log-likelihood (NLL), number of trainable model parameters, and the total training time. We compare autonomous, non-autonomous and bi-directional non-autonomous KODE models with the OT-Flow model.

Dataset	Model	# Parameters	Training time (h)	Normalized MMD	NLL
POWER (d=6)	KODE (auto)	6K	0.24	4.8e-3	6.5
	KODE (non-auto)	**12K**	**0.74**	**9.5e-4**	4.41
	KODE (bi-directional)	12K	1.70	1.7e-03	2.89
	OT-FLOW	18K	0.55	2.0e-3	**-0.10**
GAS (d=8)	KODE (auto)	8K	0.16	6.1e-3	4.61
	KODE (non-auto)	64K	1.33	2.3e-3	1.89
	KODE (bi-directional)	**32K**	**1.40**	**1.7e-3**	-1.91
	OT-FLOW	127K	2.36	3.2e-3	**-9.27**
HEPMASS (d=21)	KODE (auto)	105K	1.39	5.1e-1	29.59
	KODE (non-auto)	126K	1.40	7.0e-1	29.72
	KODE (bi-directional)	126K	1.40	4.0e-1	27.77
	OT-FLOW	**72K**	**3.45**	**2.6e-2**	**17.48**
MINIBOONE (d=43)	KODE (auto)	65K	0.15	3.7e-1	49.50
	KODE (non-auto)	86K	0.29	2.9e-1	64.78
	KODE (bi-directional)	86K	0.47	3.3e-1	48.13
	OT-FLOW	**78K**	**0.46**	**3.6e-2**	**10.65**
BSDS300 (d=63)	KODE (auto)	63K	2.89	2.7e-2	92.47
	KODE (non-auto)	252K	3.37	1.8e-2	123.73
	KODE (bi-directional)	504K	8.30	4.0e-2	77.90
	OT-FLOW	**297K**	**4.09**	**1.0e-2**	**-154.12**

HEPMASS and MINIBOONE KODE was not competitive, which is interesting since these data sets are lower-dimensional than BSDS300, suggesting that the latter data set may be high dimensional but somewhat simpler. We also found that in all of these examples the non-autonomous KODE model outperformed autonomous KODE which is not surprising given that the former has more flexibility, but this observation implies that time-varying velocity fields do lead to better performance. In the case of the BSDS300 data set it is interesting to note that autonomous KODE also achieved good

performance, which further implies that this data set is not very complex. However, for NLL metric OT-Flow significantly outperforms KODE in all benchmarks. This is likely due to the fact that the pullback of KODE is not gaussian, which we discuss below.

Figure 2.4 shows visualizations of the marginal distributions for the `GAS` data set. The top row shows 2D marginals from the target measure ν, while the second row shows corresponding marginals for the KODE samples. Visually, these marginals are very close except that the KODE marginals are slightly blurred, indicating that the finer features of the data set are not captured. The third row of Figure 2.4 shows the result of backward transport of test samples towards the Gaussian reference. Compared to the 2D benchmarks here we see that the normalizing flow is not as close to Gaussian as before. We found that this issue can be mitigated by adding an extra loss term during training which not only minimizers the MMD between the generate samples and the target, but also minimizes the MMD between the reference samples and the pull-back of the target samples. We refer to KODE models trained in this fashion as "bi-directional" KODE. The additional penalty term leads to significantly better samples in the backward transport setting as indicated in the fourth row of Figure 2.4. We also notice a significant improvement in the NLL metric across all datasets as seen in Table 2.2.

Finally, Figure 2.5 visualizes 2D marginals of the `HEPMASS` data set which was one of the examples where KODE was not competitive with OT-Flow. Here we see that while the marginals are not a perfect match, KODE appears to capture the significant structural features of the target measure ν. The weaker performance of KODE appears to be tied to the batch size used during training. Specifically with a batch size of 1000, the average MMD between two random batches from the same distribution is 4.01×10^{-1}. As the batch size increases, the MMD gradually decreases towards zero, as expected. During training, we achieve an MMD on the order of 10^{-1}, which suggests we have reached the limit of how well MMD can distinguish between two batches of the same distribution given the chosen sample size. Unfortunately, increasing the batch size further to reduce MMD is not feasible due to memory constraints. However, more efficient implementations of MMD, such as those leveraging random features [176], could allow us to use larger batch sizes and potentially improve model performance.

Across experiments, we also found that under-regularization in the RKHS norm (small λ_1)

yields good results but results in non-smooth trajectories. Over-regularization (large λ_1) hampers particle movement, leading to poor performance. In 2D experiments, regularization values below 10^{-4} produced satisfactory results, while for higher-dimensional examples, satisfactory performance is achieved with regularization below 10^{-7}.

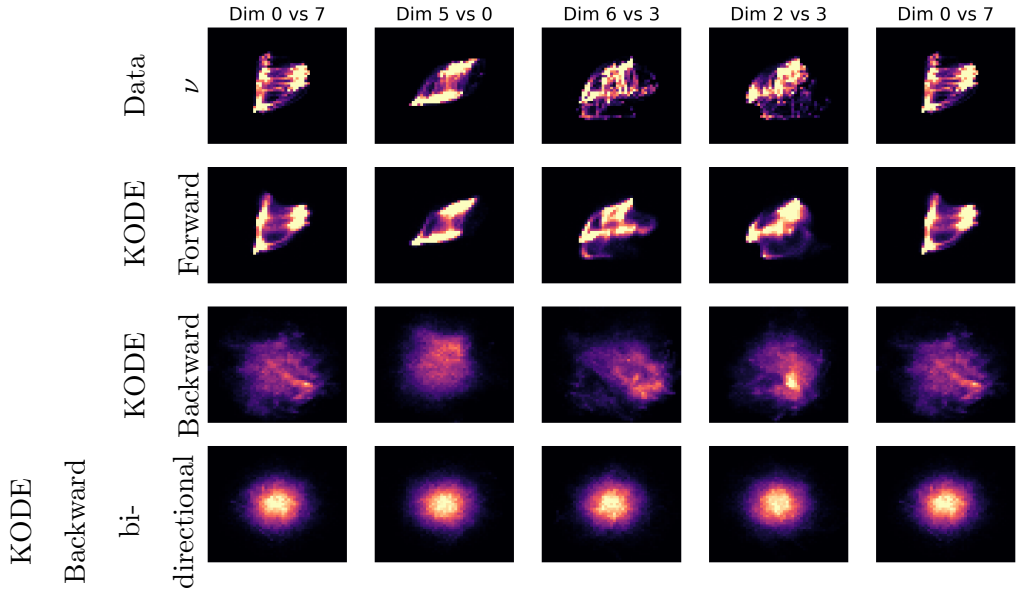

Figure 2.4: **Transport experiments on `GAS` benchmark using non-autonomous KODE.** (First row) Marginal distributions of the data measure ν. (Second row) Marginal distributions of the learned pushforward measure. (Third row) Marginal distributions of the pullback measure obtained from the backward flow of KODE. (Fourth row) Marginal distributions of the pullback measure obtained from the backward flow of KODE trained to transport η and ν between each other simultaneously.

Generative modeling for MNIST

Here we used KODE to generate images akin to the MNIST data set. Since applying KODE directly in the pixel space leads to non-satisfactory results we paired our algorithm with an intermediate autoencoder to reduce dimension of the data set. Consider an encoder $E : \mathbb{R}^{784} \to \mathbb{R}^d$ and a decoder $D : \mathbb{R}^d \to \mathbb{R}^{784}$ for MNIST such that $D(E(x)) \approx x$. If ν denotes the original target measure in

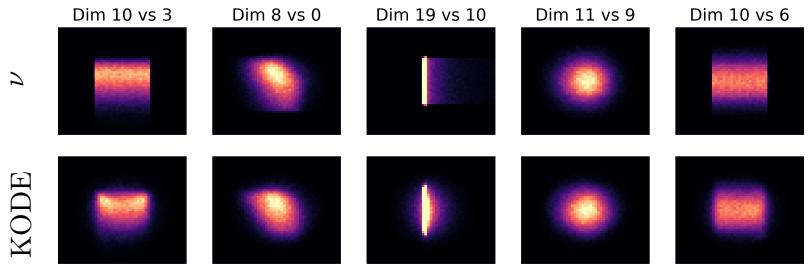

Figure 2.5: **Transport experiments on HEPMASS benchmark using non-autonomous KODE.** (First row) 2D marginals of the true data set; (Second row) 2D marginals of samples generated by KODE.

the pixel space, we aim to learn a map T using KODE such that $T\#\eta$ is close to $E\#\nu$. Once the model is trained, we can generate a new image by drawing $z \sim \eta$, and evaluating $D \circ T(z)$. To train the autoencoder we used a feedforward neural network with a dense layer for both the encoder and the decoder using ReLU activation while the output layers used a sigmoid activation function. We trained the autoencoder separately from KODE and made sure not to train to completion or to use a variational autoencoder model to ensure that KODE still had to generate samples from an interesting distribution [5]. Figure 2.6 shows the digits generated for $d = 10$. The generated samples generally resemble handwritten digits, though we do see some malformed or implausible samples. In comparison to KODE, OT-Flow produces more visually satisfactory digits. We note that in this example simply passing η through the decoder D results in images that are mostly noise and so the trained KODE model is crucial to the generator.

2.5.4 Triangular transport for conditioning

For our final set of experiments we demonstrate how KODE can be easily modified to perform likelihood-free and purely data-driven inference using the theory of triangular transport maps developed in [13]. First, we briefly describe the mathematical framework of triangular maps and how

[5]Variational autoencoders train E in such a way that $E\#\nu$ is close to a Gaussian. In this case training KODE (or any other generative model) to transform η to $E\#\nu$ is moot since an affine map would be sufficient.

Figure 2.6: **Generated MNIST digits using KODE coupled with an autoencoder.**

KODE needs to be modified and then we present an example concerning parameter estimation in a 2D benchmarks and a Lotka-Volterra ODE model with four unknown parameters.

Letting $\mathcal{Y} = \mathbb{R}^m$ and $\mathcal{U} = \mathbb{R}^d$ consider a target measure $\nu \in \mathbb{P}(\mathcal{Y} \times \mathcal{U})$. Our goal here is to obtain a generative model to sample from the conditional measure $\nu(\cdot \mid y)$ for $y \in \mathcal{Y}$. Letting $\nu_{\mathcal{Y}}$ denote the $\mathcal{Y}$-marginal of ν we consider the reference measure $\eta = \nu_{\mathcal{Y}} \otimes \eta_{\mathcal{U}} \in \mathbb{P}(\mathcal{Y} \times \mathcal{U})$ with $\eta_{\mathcal{U}} \in \mathbb{P}(\mathcal{U})$ an arbitrary measure. Finally we consider *triangular transport maps* of the form

$$T(y,u) = (y, T_{\mathcal{U}}(y,u)), \quad T_{\mathcal{Y}} : \mathcal{Y} \to \mathcal{Y}, \quad T_{\mathcal{U}} : \mathcal{Y} \times \mathcal{U} \to \mathcal{U}.$$

Then for such triangular maps we have, by [13, Thm. 2.4], that if $T\#\eta = \nu$ then $T_{\mathcal{U}}(y,\cdot)\#\eta_{\mathcal{U}} = \nu(\cdot \mid y)$. This result is the underlying principle for our modification of KODE in order to perform conditional simulation; see also the recent work [225]. More precisely, we wish to define the map T as the flow of an ODE but in such a way that the resulting transport map $T(y,u) = \phi(1,(y,u))$ is of the triangular form. This can be easily achieved by simply putting the $\mathcal{Y}$-coordinates of the velocity field v in our formula to zero. To this end, we obtain the following formulation which should be compared with Eq (2.8c):

$$\begin{cases} \underset{v \in \mathbb{V}}{\arg\min} & \mathrm{MMD}_K(\phi(1,\cdot)\#\eta^N, \nu^N) + \lambda \displaystyle\sum_{\ell=m+1}^{d+m} \|v_\ell\|_{\mathcal{V}}^2 \\ \text{s. t.} & \phi_t = v(t,\phi), \quad \phi(0,x) = x \quad \forall x \in \Omega, \\ & v(s) = (v_1(s), \ldots, v_{d+m}(s)) \\ & v_\ell = 0, \text{ for } \ell = 1, \ldots, m \\ & v_\ell = c_\ell^T V(S,\cdot), \text{ for } \ell = m+1, \ldots, d+m \end{cases} \tag{2.19}$$

To this end, the modification of KODE for conditional simulation is nearly trivial. We note that in the setting where target samples $\{(y_j, u_j)\}_{j=1}^N \sim \nu$ are given we can easily generate reference samples by forming the pairs $\{(y_j, \tilde{u_j})\}_{j=1}^N \sim \eta$ where the y_j's are copied from the target samples while the $\tilde{u}_j \sim \eta_{\mathcal{U}}$ are generated from the arbitrary reference on $\mathcal{U}$. This approach was employed in order to produce the results in Figure 2.2 by modifying the code that was used for Figure 2.1.

Parameter Inference for a Lotka-Volterra ODE

We will now use KODE for likelihood -free inference of the parameters of a Lotka-Volterra ODE, which describes the population dynamics of two interacting species using a pair of first-order non-linear ODEs; this example was used in [13] as a benchmark for a triangular GAN model for conditioning. The ODE has the form

$$\frac{dp_1}{dt} = \alpha p_1(t) - \beta p_1(t) p_2(t) \quad \frac{dp_2}{dt} = -\gamma p_2(t) + \delta p_1(t) p_2(t)$$

with initial condition $p(0) = (30, 1)$. Here P_1, P_2 denote the populations of a prey and a predator respectively. The rate of change of two populations is driven by four parameters $u = (\alpha, \beta, \gamma, \delta) \in \mathbb{R}^4$. Our goal here is to infer the values of these parameters from noisy observations of the state of the ODE.

We consider the ground truth value of the parameters $u^\dagger = (0.92, 0.05, 1.50, 0.02)$ and simulate the ODE up to time $T = 20$. The state of the ODE is then observed at time intervals of size $\Delta t = 2$, i.e., $y^\dagger_{k,i} = p_i(k\Delta t) + \xi_{k,i}$ for $i = 1, 2$ and $k = 1, \ldots, 9$ and with log-normal observation noise $\log \xi_{k,i} \sim N(0, \gamma^2)$ for $\gamma = 0.01$.

To infer the parameter u we employ Bayes' rule with a standard normal prior on u. We will use $\eta_{\mathcal{U}}$ to denote this prior since it will also be used as our reference. To generate samples from ν we proceed as follows: First draw $u_j \sim \eta_{\mathcal{U}}$ and then numerically solve the ODE with u_j and simulate the data y_j for $j = 1, \ldots, N$. This procedure generates samples from ν, the joint distribution of u, y under our prior for u and the model for the data y. Then Bayes' rule states that $\nu(\cdot \mid y^\dagger)$ is precisely the posterior distribution of u given $y^\dagger$.

We show 100,000 posterior parameter samples from KODE i.e. $T_{\mathcal{U}}(y^\dagger, w_i)$ for $w_i \sim \mathcal{N}(0, I_4)$ alongside samples from an adaptive Metropolis MCMC sampler, which we take as "ground truth", in Section 2.5.4. We generally observe similar one and two dimensional marginal distributions across

both methods. The true parameter u^* that generated the data (denoted in black) is contained in the bulk of the posterior distributions. While KODE broadly aligns with the MCMC results, it does exhibit some deviations in capturing the posterior variance. For instance, it slightly underestimates the variance in the $\delta - \beta$ marginals while overestimating it in the $\delta - \alpha$ marginals.

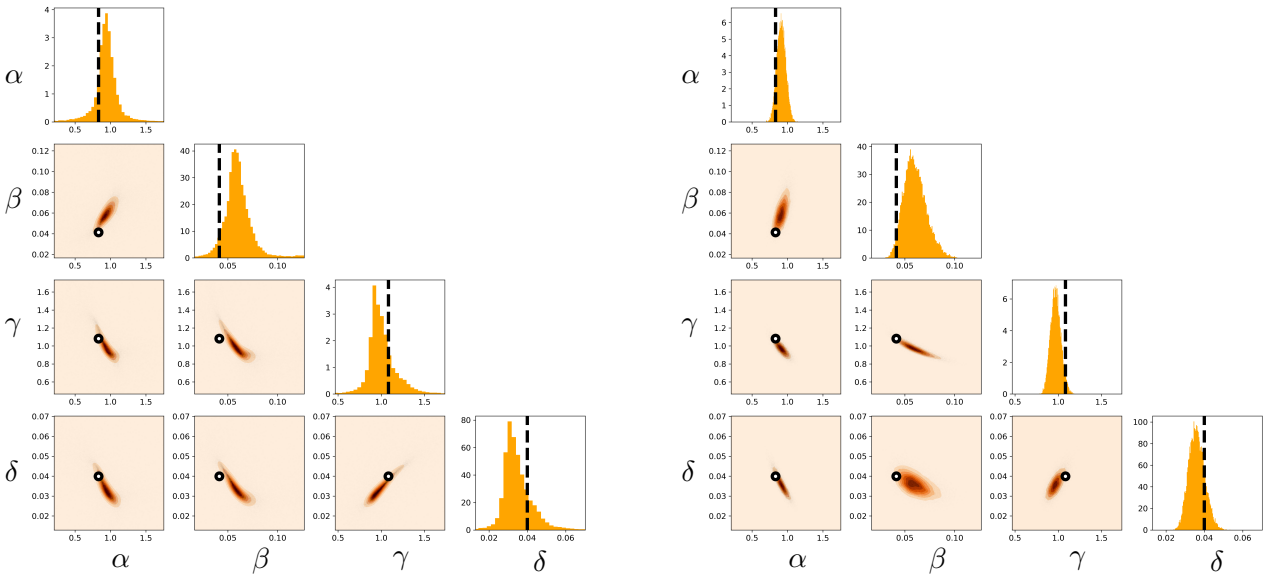

Figure 2.7: **Samples from the posterior measure of the parameters in the Lotka-Volterra model**. (Left) the triangular KODE model; (Right) the adaptive Metropolis MCMC algorithm.

2.6 Closing remarks

We introduced KODE, an approach for transport of measures with a view towards generative modeling, that was based on the theory of RKHSs and inspired by the literature on diffeomorphic matching. We presented a theoretical analysis of our model under idealized assumptions leading to quantitative error bounds in terms of the number of samples in the training data (i.e., sample complexity) as well as the complexity/degrees of freedom of our model (i.e., approximation error). To our knowlege, our theory is one few results of this kind that clearly shows the interaction between approximation error of the model and sample complexity.

We further developed algorithms based on the KODE framework that are simple and convenient to implement and mathematically interpretable. We demonstrated the effectiveness of the method on a number of benchmark transport problems in low- to high-dimensional settings and showed that KODE performs well on these benchmarks, outperforming neural net methods of a similar size in some of the benchmarks. There were also instances where KODE was not competitive.

Our results open the door to various avenues of research in both theory and algorithms. From a theory standpoint, it would be interesting to close the gap between our theoretical framework and the implemented version of KODE: (1) it is interesting to further characterize the relationship between our constrained formulation of KODE Eq (2.8c) and its unconstrained version Eq (2.15) which is implemented; (2) it is interesting to extend our error bounds to account for the error of numerical ODE solver that is used in the model; (3) finally, it is interesting to consider KODE with discrepancies besides MMD and try to obtain error bounds in that case. From an algorithmic standpoint, it is interesting to investigate how the performance of KODE can be improved: (1) our current implementation is very sensitive to the choice of the inducing points which may also scale badly with dimension. We rely on k-means clustering to select representative inducing points that summarize the empirical data support. However, k-means struggles with the curse of dimensionality, making it difficult to effectively capture the structure of high-dimensional data. Therefore it is interesting to investigate other strategies such as random features; (2) our formulation is also sensitive to the choice of the kernel in the MMD term and so a strategy for choosing that kernel to maximally differentiate the target and generated samples is of interest; (3) while we currently employ stochastic gradient descent for the training of KODE it is interesting to design Newton-type algorithms that can leverage the quadratic structure of our RKHS penalty terms to achieve faster convergence and potentially avoid undesirable local minima despite the non-convex setting [139].

Chapter 3

AN ERROR ANALYSIS FRAMEWORK FOR MINIMUM DIVERGENCE TRANSPORT MODELS WITH APPLICATION TO GENERATIVE MODELS

3.1 Prelude to the chapter

Understanding the theoretical underpinnings of generative models is essential for advancing their development and ensuring reliable performance across diverse applications. Although the empirical successes of these models in fields such as language modeling, image synthesis, and scientific computing are well documented, their theoretical foundations, particularly the interplay between model complexity, sample size, and generalization, remain less understood. This chapter aims to address this gap by developing a quantitative framework for analyzing the generalization properties of minimum divergence transport (MDT) models.

Building on the theoretical results from earlier chapters, we derive rigorous error bounds that decompose the generalization error into approximation and stochastic components. These bounds depend on the choice of the hypothesis class $\mathcal{S}$ for the transport map and the divergence D used to quantify discrepancies between measures. Through this analysis, we identify key factors influencing the trade-offs between approximation bias, which reflects how well the model class can represent the target, and stochastic variance, which accounts for the noise introduced by finite samples.

This chapter is structured as follows. In Section 2.3, we introduce the general framework for analyzing generalization error in MDT models, detailing the theoretical foundations and presenting master theorems that apply across a wide range of divergences and hypothesis classes. Section 3.4 extends these results to specific discrepancies, including IPMs, Wasserstein distances, and FUSE discrepancies, and demonstrates their implications for parametric and nonparametric models. Finally, in Section 4.3.4, we validate these theoretical findings through a comprehensive suite of numerical experiments, highlighting the behavior of RKHS-based models, neural networks, and polynomial transport maps in different error regimes.

The contents of this chapter are in preparation for submission.

3.2 Introduction

Generative modeling is a fundamental unsupervised learning task in machine learning (ML) with wide applications in language models, computer vision, scientific computing, and inference. A fundamental theoretical question at the heart of generative models is the characterization of their generalization properties, simply put, *can we quantify how good a generative model is in terms of the complexity of the model class at hand, e.g., size of a neural network architecture, and the size of the training data, i.e., sample complexity?*

In recent years, a lot of attention has been dedicated to answering the above questions in the context of specific models for density estimation such as adversarial training losses [131, 213, 212, 106, 200] or optimal transport [227, 199]. In this article, we build upon these previous works and provide a series of theoretical results, supported by numerical experiments, that provide quantitative error bounds for the class of minimum divergence transport (MDT) models [165, 19, 14].

Broad families of generative models, most notably, generative adversarial networks (GANs) [91, 129] and normalizing flows [181] belong to the MDT class. More precisely, these are models trained by solving the optimization problems of the form

$$\widehat{\nu}^N := \widehat{T}^N \# \eta, \quad \widehat{T}^N := \underset{T \in \mathcal{S}}{\arg\min}\, D(T\#\eta, \nu^N), \tag{3.1}$$

where $\eta \in \mathbb{P}(\Omega)$ is a reference Borel probability measure [1] on a domain $\Omega \subseteq \mathbb{R}^d$ and $\nu \in \mathbb{P}(\Omega)$ [2] is a target probability measure with ν^N denoting its empirical approximation obtained from N independent and identically distributed (i.i.d.) samples; thus ν^N constitutes our training data in the context of generative models. $\mathcal{S}$ is a hypothesis/model class for generative models. For any fixed transport map $T : \Omega \to \Omega$, we write $T\#\eta$ to denote the pushforward of η, i.e., the law of $T(z)$ for $z \sim \eta$. Finally, $D : \mathbb{P}(\Omega) \times \Omega \to \mathbb{R}_{\geq 0}$ denotes a statistical discrepancy or divergence, that measures the dissimilarity between two probability measures.

---

[1] In most applications one simply chooses standard normal random variables.

[2] Here we take η, ν as measures on the same set Ω for simplicity. Our results can easily generalize to the case where these measures are defined on different sets or spaces of different distributions as is often done in many generative models.

3.2.1 Main Contributions

Our main theoretical result states that under relatively mild assumptions on D, one can obtain a quantitative generalization bound for $\widehat{T}^N$ in terms of the approximation bias of $\mathcal{S}$, and the divergence between ν^N and the true distribution ν. We distill this result in the form of the following theorem that follows from our analysis in Section 2.3.

Theorem 3.1. *Consider $\Omega \subseteq \mathbb{R}^d$ and Borel probability measures $\eta, \nu \in \mathbb{P}(\Omega)$. Let $D : \mathbb{P}(\Omega) \times \mathbb{P}(\Omega) \to \mathbb{R}_{\geq 0}$ be a statistical discrepancy satisfying:*

(i) (Near triangle property) There exist discrepancies D' and D'' such that $D(\mu_1, \mu_2) \leq D''(\mu_1, \mu_3) + D''(\mu_3, \mu_2)$ and $D'(\mu_1, \mu_2) \leq D(\mu_1, \mu_3) + D(\mu_3, \mu_2)$ for all triples $\mu_1, \mu_2, \mu_3 \in \mathbb{P}(\Omega)$.

(ii) (Map stability) There exists a constant $C_\eta \geq 0$ (possibly depending on η) such that $D''(T\#\eta, T'\#\eta) \leq C_\eta \|T - T'\|_{L^2_\eta(\Omega;\Omega)}$ for any pair of maps $T, T' \in L^2_\eta(\Omega;\Omega)$ [3].

Then for any ground truth map $T^\dagger \in L^2_\eta(\Omega;\Omega)$ satisfying $T^\dagger\#\eta = \nu$, and empirical estimator $\widehat{T}^N$ given by Eq (3.1) with a model class $\mathcal{S} \subseteq L^2_\eta(\Omega;\Omega)$, it holds that

$$D'(\nu, \widehat{\nu}^N) \leq C \left(\inf_{T \in \mathcal{S}} \|T - T^\dagger\|_{L^2_\eta(\Omega;\Omega)} + D''\tau(\gamma)(\nu, \nu^N) + D''(\nu^N, \nu) \right), \tag{3.2}$$

where $C > 0$ is a constant.

We note that the error bound above is with regards to the weaker metric D' instead of the original metric D which we minimize over. However, for discrepancies which satisfy the weak triangle inequality, D', D'' and D are equivalent. The first term in the bound above characterizes the approximation error of the model class $\mathcal{S}$ for the ground truth map $T^\dagger$. We emphasize that in general $T^\dagger$ is not unique but our bound holds for any choice of this ground truth map and so, surprisingly, the MDT map $\widehat{T}^N$ automatically inherits the optimal approximation rate corresponding to the most regular choice of $T^\dagger$. The remaining terms in the bound are entirely independent of the generative model and depend only on the measure ν and the choice of D. In practical settings and for particular algorithms, the terms in our bound can be made quantitative using off-the-shelf

[3] Here $L^2_\eta(\Omega;\Omega)$ denotes the Lebesgue space of equivalence classes of functions on Ω that are square integrable with respect to η.

results from approximation theory [216] for the first term, and existing sample complexity rates for various divergences such as those in [131, 74]. In Section 3.4 we give numerous applications of the above theorem for minimum maximum mean discrepancy (MMD) transport maps in reproducing kernel Hilbert spaces (RKHSs), GAN losses with neural net maps, and Wasserstein distances with polynomial maps. Finally, we note that the bound Eq (3.2) can be extended to the case where $\widehat{T}^N$ is obtained by solving Equation (3.1) with η replaced with an empirical approximation as well.

Thus, our main contributions can be summarized as follows:

- We introduce a master theorem for obtaining generalization bounds for MDT generative models under appropriate assumptions as summarized in Section 3.3
- We apply our results to various instances of MDT models in Section 3.4.
- We present synthetic numerical experiments for some of the aforementioned MDT models that test the sharpness of our rates and go beyond our theoretical analysis in Section 3.5.

3.2.2 Relevant literature

The closest work to our analysis is the seminal paper [131] where minimax optimal generalization bounds were obtained for minimum divergence generative models with D taken to be an integral probability metric (IPM). Indeed, one of the key results Theorem 3.3, is a generalization of the oracle bound [131, Lem. 23] from IPMs to more general discrepancies. The more recent works [106, 213, 212] give refined minimax bounds under further assumptions on the target measure ν such as low-dimensional support, invariance under certain group actions, and smoothness of the densities. An important distinction in our work is that we focus on transport models, since most generative models fall in this category, and state our bounds in terms of the approximation error of the ground truth map $T^\dagger$ while the assumptions in the aforementioned works often concern the density of the target measure ν. While the regularity of $T^\dagger$ and the density of ν are related (see for example [124]) the approximation properties of the two objects may be very different; see our discussion in SM.

3.3 Theoretical Results

Here we collect our main theoretical results leading up to and extending Theorem 3.1.

3.3.1 Set up

Consider $\Omega \subseteq \mathbb{R}^d$ and Borel probability measures $\eta, \nu \in \mathbb{P}(\Omega)$. Let $\mathcal{T}$ be a Banach space of measurable transport maps from Ω to itself and let $\mathcal{S} \subset \mathcal{T}$ be the model class (may be parametric or not) as in Equation (3.1) that is closed in $\mathcal{T}$. Let the statistical discrepancy D be a map $D : \mathbb{P}(\Omega) \times \mathbb{P}(\Omega) \to \mathbb{R}_{\geq 0}$ such that $D(\mu, \mu) = 0$.

3.3.2 A generic bound for a family of discrepancies

We give several abstract theoretical results for the family of discrepancies that satisfy the following property.

Definition 3.2 (Near-Triangle Property)**.** *Let Γ be an index set. Consider a family of discrepancies $\mathcal{D}_\Gamma = \{D_\gamma \mid \gamma \in \Gamma\}$ defined for elements of $\mathbb{P}(\Omega)$. We say that the family $\mathcal{D}_\Gamma$ satisfies a near-triangle property if there exists a map $\tau : \Gamma \to \Gamma$ such that for all $\gamma \in \Gamma$ and any triple $\mu_1, \mu_2, \mu_3 \in \mathbb{P}(\Omega)$, we have*

$$D_\gamma(\mu_1, \mu_2) \leq C_\gamma \left(D_{\tau(\gamma)}(\mu_1, \mu_3) + D_{\tau(\gamma)}(\mu_3, \mu_2) \right) \tag{3.3}$$

with $C_\gamma \geq 0$ independent of μ_i.

Our main motivation for considering the families $\mathcal{D}_\Gamma$ is the so-called class of FUSE discrepancies which we will define in the parlance of [27]: Let $\mathcal{D}_I := \{D_i | i \in I\}$ be a family of discrepancies over $\mathbb{P}(\Omega)$, parameterized by $i \in I$, each satisfying the weak triangle inequality, i.e., $D_i(\mu_1, \mu_2) \leq C_i(D_i(\mu_1, \mu_3) + D_i(\mu_3, \mu_2))$ where $C_i \geq 0$. Then, for some distribution $\varrho \in \mathbb{P}(I)$ and parameter $\gamma \geq 0$, the FUSE discrepancy over the $\mathcal{D}_I$ is defined as

$$\text{FUSE}\gamma(\mu, \mu') := \frac{1}{\gamma} \log \left(\mathbb{E}_{i \sim \varrho} \exp(\gamma D_i(\mu, \mu')) \right). \tag{3.4}$$

This definition creates a soft maximum over the family of discrepancies $\mathcal{D}_I$, controlled by the parameter γ. As $\gamma \to \infty$, FUSE_γ approaches $\operatorname{ess\,sup}_{i \in \operatorname{supp} \varrho} \mathcal{D}_i(\mu, \mu')$ where $\operatorname{supp} \varrho \subseteq I$ denotes the support of ϱ, while for finite γ, it provides a smooth approximation. Conversely, when $\gamma \to 0$ it

approximates the expectation $\mathbb{E}_{i\sim\varrho}D_i(\mu,\mu')$. As we will show in Section 3.4.3, the family $\mathrm{FUSE}_\Gamma := \{\mathrm{FUSE}_\gamma\}_{\gamma\geq 0}$ has the near-triangle property defined above. The FUSE discrepancy generalizes the so-called MMD-FUSE loss from [27], where D_i were taken to be MMD defined from a parametric family of kernels; see Section 3.4.1 for details. The generalized FUSE loss is significant as it allows us to combine discrepancies that may or may not be related. Furthermore, we emphasize that we can always replace $\mathcal{D}_\Gamma$ with a single discrepancy D which allows us to generalize our main error bounds to simple minimum divergence estimators of transport maps.

With the above notions at hand, we can now state and prove our main theoretical results, a master theorem that decomposes the generalization error of minimum divergence estimators in terms of an approximation bias and an stochastic error due to training data set.

Theorem 3.3. *Let $\mathcal{D}_\Gamma$ be a family of discrepancies with the near-triangle property as in Definition 3.2. For any $D_\gamma \in \mathcal{D}_\Gamma$, suppose*

$$\widehat{T}^N = \operatorname*{arg\,min}_{T\in\mathcal{S}} D_\gamma(T\#\eta,\nu^N), \quad \widehat{T} = \operatorname*{arg\,min}_{T\in\mathcal{S}} D_\gamma(T\#\eta,\nu).$$

Then, for any $\gamma' \in \tau^{-1}(\gamma)$ (the pre-image of the singleton $\{\gamma\}$) it holds that

$$D_{\gamma'}(\widehat{T}^N\#\eta,\nu) \leq C\left(D_{\tau(\gamma)}(\widehat{T}\#\eta,\nu) + D_{\tau(\gamma)}(\nu,\nu^N) + D_{\tau(\gamma)}(\nu^N,\nu)\right) \tag{3.5}$$

where $C>0$ depends on C_γ and $C_{\gamma'}$.

Proof. Using the near-triangle property, the optimality of $\widehat{T}^N$ and the near-property property, and the near-triangle property once more, we can write

$$\begin{aligned} D_{\gamma'}(\widehat{T}^N\#\eta,\nu) &\leq C_{\gamma'}\left(D_\gamma(\widehat{T}^N\#\eta,\nu^N) + D_\gamma(\nu^N,\nu)\right) \\ &\leq C_{\gamma'}\left(D_\gamma(\widehat{T}\#\eta,\nu^N) + C_\gamma\left[D_{\tau(\gamma)}(\nu^N,\nu^N) + D_{\tau(\gamma)}(\nu^N,\nu)\right]\right) \\ &\leq C_{\gamma'}\left(C_\gamma\left[D_{\tau(\gamma)}(\widehat{T}\#\eta,\nu) + D_{\tau(\gamma)}(\nu,\nu^N) + D_{\tau(\gamma)}(\nu^N,\nu)\right]\right) \end{aligned}$$

□

The bound Eq (3.5) separates the approximation bias of $\mathcal{S}$ in the first term from the empirical error due to random samples in the training data in the remaining terms; generalizing the oracle inequality proved for IPMs in [131, Lem. 23]. While the second and third terms are often controlled using empirical process theory/concentration bounds [131, 200, 213], the first term can be controlled using techniques from approximation theory [216, 14] under further assumptions.

Definition 3.4. *We say that the triple $(\mathcal{T}, D, \eta)$ are Lipschitz stable if*

$$D(T\#\eta, T'\#\eta) \leq C_\eta \|T - T'\|_{\mathcal{T}}, \quad \forall T, T' \in \mathcal{T}$$

where $C_\eta \geq 0$ is a constant that is independent of T and T'[4]*. Moreover, we say that the triple are locally Lipschitz stable if the bound holds for a constant $C_\eta(\mathcal{S}) \geq 0$ for all $T, T' \in \mathcal{S} \subset \mathcal{T}$.*

Note that one may also extend this definition to a α-Hölder-stabilty assumption by asking for $D(T\#\eta, T'\#\eta) \leq C_\eta \|T - T'\|_{\mathcal{T}}^\alpha$ for some exponent $\alpha \geq 0$ but we will not pursue this here as the Lipschitz case is sufficient for our applications. Indeed we will show in Section 3.4 that Lipschitz stability can be easily verified in many common settings such as minumum MMD, Wasserstein, or GAN models.

Naturally if $(\mathcal{T}, D_{\tau(\gamma)}, \eta)$ are Lipschitz stable and assuming that there exists a map $T^\dagger \in \mathcal{T}$ satisfying $T^\dagger\#\eta = \nu$, we immediately obtain the bound $D_{\tau(\gamma)}(T\#\eta, \nu) = D_{\tau(\gamma)}(T\#\eta, T^\dagger\#\eta) \leq C_\eta \|T - T^\dagger\|_{\mathcal{T}}$. Using this calculation we can obtain an extension of Theorem 3.1 which we state below for the globally Lipschitz case, noting that it trivially holds also for to the locally Lipschitz case:

Theorem 3.5. *Suppose D_γ belongs to $\mathcal{D}_\Gamma$ that satisfies the near-triangle property and $(\mathcal{T}, D_{\tau(\gamma)}, \eta)$ are Lipschitz stable. Let $T^\dagger \in \mathcal{T}$ be any map satisfying $T^\dagger\#\eta = \nu$ and let $\widehat{T}^\dagger := \arg\min_{T \in \mathcal{S}} \|T - T^\dagger\|_{\mathcal{T}}$ be its best approximation in $\mathcal{S}$.* [5] *Then, for any $\gamma' \in \tau^{-1}(\gamma)$, we have the bound*

$$D_{\gamma'}(\widehat{T}^N\#\eta, \nu) \leq C \left(\|\widehat{T}^\dagger - T^\dagger\|_{\mathcal{T}} + D_{\tau(\gamma)}(\nu, \nu^N) + D_{\tau(\gamma)}(\nu^N, \nu) \right) \tag{3.6}$$

for a constant $C > 0$ that depends only on $C_\gamma, C_{\gamma'}$ and C_η.

Remark 3.6. *While we state our results with Ω being a finite-dimensional Euclidean space, this assumption is not needed. We can generalize these results assuming Ω is a Banach space.*

When D is a metric, our result generalizes by taking $\mathcal{D}_\Gamma$ to be the singleton $\{D\}$. Several metrics like Wasserstein distance, MMD, and certain GAN losses fall within this category. In addition, it

[4]While this constant may depend on the entire triple $(\mathcal{T}, \mathcal{D}, \eta)$, we highlight only its dependence on η since in many practical applications η is the main object of interest that also influences the choice of the norm on $\mathcal{T}$.

[5]Note that the arg min is well-defined due to the assumption that $\mathcal{S}$ is closed in $\mathcal{T}$.

follows that $D(\nu, \nu^N) = D(\nu^N, \nu)$ from the symmetry property. This leads to the following bound for metrics which we summarize for convenience:

Corollary 3.7. *Suppose D is a metric and $(\mathcal{T}, D, \eta)$ are Lipschitz stable. Let $T^\dagger \in \mathcal{T}$ be any map satisfying $T^\dagger \# \eta = \nu$ and let $\widehat{T}^\dagger := \arg\min_{T \in \mathcal{S}} \|T - T^\dagger\|_{\mathcal{T}}$ be its best approximation in $\mathcal{S}$. Then, it holds that*

$$D(\widehat{T}^N \# \eta, \nu) \leq C \left(\|\widehat{T}^\dagger - T^\dagger\|_{\mathcal{T}} + D(\nu, \nu^N) \right), \tag{3.7}$$

for a constant $C > 0$ that depends only on C_η.

3.4 Example Applications

We give a collection of results, first focusing on single divergences of certain types, namely IPMs and OT. Then, we generalize these to various FUSE models based on these classes. Throughout this section, we assume that $T^\dagger \in \mathcal{T}$ exists and $\widehat{T}^\dagger$ is defined as in Theorem 3.5.

3.4.1 IPMs

For our first application, we consider the case where D is an IPM. This setting for generative models has been studied extensively in the literature; see for example [131, 200, 37] and Section 3.2.2. Given a set $\mathcal{F}$ of real valued functions on Ω, we define the IPM

$$D_{\mathcal{F}}(\mu, \mu') := \sup_{f \in \mathcal{F}} \left(\mathbb{E}_{x \sim \mu} f(x) - \mathbb{E}_{x' \sim \mu'} f(x') \right)$$

The IPM class is attractive since it includes many common probability metrics including: Total variation distance when $\mathcal{F}$ is the space of bounded and measurable functions; Wasserstein-1 distance when $\mathcal{F}$ is the space of 1-Lipschitz functions; and MMD when $\mathcal{F}$ is the unit ball of an RKHS; and finally GANs when $\mathcal{F}$ is a parametric family of neural nets.

For a function $\phi : \Omega \times \Omega \to \mathbb{R}_{\geq 0}$, we write $\mathrm{Lip}_\phi := \left\{ f : \Omega \to \mathbb{R} \mid \frac{|f(x) - f(x')|}{|x - x'|} \leq \phi(x, y) \right\}$. We further write $\mathrm{Lip}_{\phi,B} \subset \mathrm{Lip}_\phi$ to denote the subset of ϕ-Lipschitz functions that are point-wise bounded by a constant $B > 0$, i.e., $\sup_{x \in \Omega} |f(x)| \leq B$. Moreover, given the empirical measure ν^N, we write $\widehat{\mathbb{R}}(\mathcal{F}; \nu^N)$ to denote the empirical Rademacher complexity of $\mathcal{F}$ over the empirical samples defining ν^N. Explicitly, for a set of i.i.d. samples $\{y_i\}_{i=1}^N$, the empirical Rademacher complexity is

defined as

$$\widehat{\mathbb{R}}(\mathcal{F}; \{y_i\}_{i=1}^N) := \mathbb{E}_\sigma \sup_{f \in \mathcal{F}} \left| \frac{1}{N} \sum_{i=1}^N \sigma_i f(y_i) \right|,$$

where σ_i are Rademacher random variables with $\mathbb{P}(\sigma_i = +1) = \mathbb{P}(\sigma_i = -1) = \frac{1}{2}$. Finally let us write $\mathbb{P}_{\mathcal{F}}(\Omega) := \{\mu \in \mathbb{P}(\Omega) \mid \mathbb{E}_{x\sim\mu} f(x) < +\infty, \quad \forall f \in \mathcal{F}\}$. We can readily verify that IPMs are symmetric and satisfy the triangle inequality for measures in $\mathbb{P}_{\mathcal{F}}(\Omega)$. It remains for us to verify that IPMs are lipschtiz stable:

Lemma 3.8. *Let $D_{\mathcal{F}}$ be an IPM with $\mathcal{F} \subseteq Lip_\phi$. Suppose $\mathcal{T} \subseteq L^p_\eta(\Omega;\Omega)$ for some $p \in [1, +\infty]$, and $\nu \in \mathbb{P}_{\mathcal{F}}(\Omega)$. Assume further that $\phi \circ (T(\cdot), T'(\cdot)) \in L^q_\eta(\Omega;\Omega)$ for all $T, T' \in \mathcal{T}$ and $q = \frac{p}{1-p}$. Then, it holds that*

$$D_{\mathcal{F}}(T\#\eta, T'\#\eta) \leq \|\phi(T(\cdot), T'(\cdot))\|_{L^q_\eta(\Omega;\Omega)} \|T - T'\|_{L^p_\eta(\Omega;\Omega)} \tag{3.8}$$

Proof. Using the Lipschitz property, we have

$$\begin{aligned} D_{\mathcal{F}}(T\#\eta, T'\#\eta) &= \sup_{f\in\mathcal{F}} \int_\Omega f(T(x)) - f(T'(x)) d\eta(x) \\ &\leq \sup_{f\in\mathcal{F}} \int_\Omega \left| f(T(x)) - f(T'(x)) \right| d\eta(x) \\ &\leq \int_\Omega \phi(T(x), T'(x)) |T(x) - T'(x)| d\eta(x) \end{aligned}$$

The desired result follows from Hölder's inequality. □

Strengthening our assumptions on the class $\mathcal{F}$ will further allow us to obtain a quantitative error bound for minimum IPM estimators as follows:

Proposition 3.9. *Let $\mathcal{F} \subseteq \mathrm{Lip}_{\phi,B}$, and both $\mathcal{T}$ and ϕ satisfy the conditions in Lemma 3.8. Let $\widehat{T}^N$ be the solution to Eq (3.1) with $D \leftarrow D_{\mathcal{F}}$. Then with probability $1 - \delta > 0$ it holds that*

$$\begin{aligned} D_{\mathcal{F}}(\widehat{T}^N\#\eta, \nu) \leq \|\phi(\widehat{T}^\dagger(\cdot), T^\dagger(\cdot))\|_{L^q_\eta(\Omega;\Omega)} \|\widehat{T}^\dagger - T^\dagger\|_{L^p_\eta(\Omega;\Omega)} + 2\widehat{\mathbb{R}}(\mathcal{F}; \nu^N) \\ + \sqrt{\frac{18B^2}{N} \log\left(\frac{2}{\delta}\right)} \end{aligned}$$

Proof. First, we verify that $(\mathcal{T}, D_{\mathcal{F}}, \eta)$ is locally Lipschitz stable around $T^\dagger$ using Lemma 3.8. Moreover, $D_{\mathcal{F}}$ satisfies the triangle inequality and symmetry. Using Theorem 3.5, we can decompose

the generalization error into approximation error and the stochastic error. Finally, we control the latter in terms of the empirical Rademacher complexity of $\mathcal{F}$ using the master bounds in [205, Thm 11]. □

MMD

As a first application of Proposition 3.9 we consider the case where $\mathcal{F}$ is the unit ball in an RKHS in which case $\mathcal{D}_{\mathcal{F}}$ is called the MMD; see [24, 157] for a review of RKHS theory. More precisely, let $K : \Omega \times \Omega \to \mathbb{R}$ be a Mercer kernel and write $\mathcal{H}_K$ to denote its RKHS with norm $\|\cdot\|_K$. Write $\mathbb{P}_K(\Omega) := \{\mu \in \mathbb{P}(\Omega) \mid \mathbb{E}_{x\sim\mu} K(x,x) < +\infty\}$. We then define the MMD based on the kernel K as $\mathrm{MMD}_K(\mu,\mu') := \mathcal{D}_{\mathcal{F}}(\mu,\mu')$ for all $\mu,\mu' \in \mathbb{P}_K(\Omega)$ with $\mathcal{F} = \{f \in \mathcal{H}_K \mid \|f\|_K \leq 1\}$; see [147] for a review of MMD. We can then apply Proposition 3.9 to this setting, along with known bounds on the Rademacher complexity of RKHS balls [205, Corollary 12(ii)] to obtain the following corollary.

Corollary 3.10. *Let $K : \Omega\times\Omega \to \mathbb{R}$ be a Mercer kernel satisfying $\|K(x,\cdot) - K(x',\cdot)\|_K \leq L|x-x'|$ for a constant $L > 0$, suppose $\mathcal{T} \subseteq L^p_\eta(\Omega;\Omega)$ for $p \in [1,+\infty]$, and $\nu \in \mathbb{P}_K(\Omega)$. Let $\widehat{T}^N$ be the solution to Eq (3.1) with $D \leftarrow \mathrm{MMD}_K$. Then with probability $1-\delta > 0$ it holds that*

$$\mathrm{MMD}_K(\widehat{T}^N\#\eta,\nu) \leq L\|\widehat{T}^\dagger - T^\dagger\|_{L^p_\eta(\Omega;\Omega)} + \sqrt{\frac{1}{N}\sup_{x\in\Omega} K(x,x)}\left(4 + \sqrt{18\log\left(\frac{2}{\delta}\right)}\right).$$

GANs

Next we consider the case where $\mathcal{F}$ is a parametric neural net class in which case Equation (3.1) resembles the training of a GAN. This formulation of GANs is widely studied in the literature; see for example [131, 106] as well as our literature review in Section 3.2.2.

Consider feed-forward neural nets of the form

$$f(x) = \psi \circ (A_l\sigma \circ A_{l-1}\sigma \circ \ldots A_2\sigma \circ A_1 x),$$

where $A_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ are weight matrices with $d_0 = d$ (the dimension of Ω), σ are the inner activation functions, and ψ is the output activation function. We write $\mathcal{F}_{\mathrm{NN}}(l,M,B)$ to denote the space of such neural nets with l layers and such that the weight matrices satisfy $\|A_\ell\|_F \leq M$, σ and ψ are both 1-Lipschitz and ψ is also bounded, i.e., $\sup_{y\in\mathbb{R}^{d_l}} |\psi(y)| \leq B$. A simple example of such a

network is when σ is a ReLU activation while ψ is a sigmoid. We then define D_{GAN} to be the IPM defined via $\mathcal{F}_{\mathrm{NN}}(l, M, B)$.

Since the activation functions are 1-Lipschitz, we infer that the elements of $\mathcal{F}_{\mathrm{NN}}(l, M, B)$ are uniformly M^l-Lipschitz. The boundedness of ψ also implies that the neural nets are also bounded. This allows us to apply Proposition 3.9 together with a bound on the Rademacher complexity of neural nets in [87] to obtain the following error bound for GANs.

Corollary 3.11. *Suppose Ω is bounded, $\mathcal{T} \subseteq L^p_\eta(\Omega;\Omega)$ for $p \in [1, +\infty]$, and $\widehat{T}^N$ is the solution to Eq (3.1) with $D \leftarrow \mathcal{D}_{\mathrm{GAN}}$ defined over the neural net discriminator class $\mathcal{F}_{\mathrm{NN}}(l, M, B)$. Then with probability $1 - \delta > 0$ it holds that*

$$\begin{aligned}\mathcal{D}_{\mathrm{GAN}}(\widehat{T}^N \# \eta, \nu) \leq M^l \|\widehat{T}^\dagger - T^\dagger\|_{L^p_\eta(\Omega;\Omega)} \\ + \left(2\mathrm{diam}(\Omega)(\sqrt{2\log(2)l}+1)M^l + \sqrt{18B^2\log\left(\frac{2}{\delta}\right)}\right)\sqrt{\frac{1}{N}}\,.\end{aligned}$$

3.4.2 Optimal transport distances

Let us now consider the case of OT distances, in particular, the standard Wasserstein-p metrics defined as

$$W_p(\mu, \mu') := \left(\inf_{\pi \in \Pi(\mu,\mu')} \mathbb{E}_{(x,x')\sim\pi}|x - x'|^p\right)^{1/p},$$

with $\Pi(\mu, \mu') \in \mathbb{P}(\Omega \times \Omega)$ denoting the set of couplings between μ, μ', i.e., probability measures on $\Omega \times \Omega$ whose first marginals match μ and the second marginals match μ'; see our review in Section 3.2.2. This metric is finite for measures in the set $\mathbb{P}_p(\Omega) := \{\mu \in \mathbb{P}(\Omega) \mid \mathbb{E}_{x\sim\mu}|x|^p < +\infty\}$ and taken to be infinite otherwise.

A straightforward calculation (refer to [14, Thm. 3.1]) yields that the triples $(L^q_\eta(\Omega;\Omega), W_p, \eta)$ are Lipschitz stable for any $\eta \in \mathbb{P}_p(\Omega)$ and $q \geq p$, indeed, we have the bound

$$W_p(T\#\eta, T'\#\eta) \leq \|T - T'\|_{L^q_\eta(\Omega;\Omega)}.$$

Combining this bound with Equation (3.5) and known bounds on $W_p(\nu, \nu^N)$ such as those in [74] allows us to obtain our first error bound for minimum W_p transport models.

Proposition 3.12. *Suppose $\eta, \nu \in \mathbb{P}_p(\Omega)$ for $p \geq 1$ and $\mathcal{T} \subseteq L^q_\eta(\Omega;\Omega)$ for $q \geq p$. Let $\widehat{T}^N$ be the solution to Eq (3.1) with $D \leftarrow W_p$. Then with probability $1-\delta > 0$ it holds that*

$$W_p(\widehat{T}^N \# \eta, \nu) \leq \|\widehat{T}^\dagger - T^\dagger\|_{L^q_\eta(\Omega;\Omega)} + \frac{C}{\delta} \cdot \begin{cases} N^{-1/2p} & d < 2p, \\ N^{-1/2p} \log(1+N)^{1/p} & d = 2p, \\ N^{-1/d} & d > 2p, \end{cases}$$

where $C(p,d,\nu) > 0$ is a constant and we recall d is the dimension of the domain Ω.

The bound in the above theorem is theoretically interesting but it is not practical since solving Equation (3.1) with Wasserstein distances is not feasible. Instead one may consider various relaxations of these distances that lead to more convenient formulations. As an example of such a case we now consider the sliced variation.

Sliced wasserstein distances

First introduced in [175, 33], the Sliced Wasserstein distance between two measures is defined as the mean Wasserstein distance between their one-dimensional projections. Let $\mathbb{S}^{d-1} = \{\theta \in \mathbb{R}^d : \|\theta\| = 1\}$ and take $\pi_\theta : \Omega \to \mathbb{R}$ to be the projection $\pi_\theta(x) = x^\intercal \theta$ for all $\theta \in \mathbb{S}^{d-1}$. Let ϱ denote any probability measure that is absolutely continuous with respect to the Haussdorf measure on $\mathbb{S}^{d-1}$. Following [33, 34], the p-th order Sliced Wasserstein distance between two measures $\mu, \mu' \in \mathbb{P}_p(\Omega)$ is defined as

$$\mathrm{SW}_p(\mu, \mu') = \left(\int_{\mathbb{S}^{d-1}} W_p^p(\pi_\theta \# \mu, \pi_\theta \# \mu') d\varrho(\theta) \right)^{1/p}$$

Observe that the Sliced Wasserstein distance inherits the triangle inequality and symmetry properties of the standard Wasserstein distance. We can readily verify that the triples $(L^q_\eta(\Omega;\Omega), \mathrm{SW}_p, \eta)$ are Lipschitz stable for any $\eta \in \mathbb{P}_p(\Omega)$ and $q \geq p$. From a straightforward calculation (see [14, Thm. 3.1]), we get the bound $W_p^p(\pi_\theta \circ T \# \mu, \pi_\theta \circ T \# \mu') \leq \|\theta\|^p \|T - T'\|_{L^q_\eta(\Omega;\Omega)} = \|T - T'\|_{L^q_\eta(\Omega;\Omega)}$. This immediately tells us that

$$\mathrm{SW}_p(T \# \eta, T' \# \eta) \leq \|T - T'\|_{L^q_\eta(\Omega;\Omega)}.$$

Then, the application of Theorem 3.5 and known empirical rates for $\mathrm{SW}_p(\nu, \nu^N)$ from [136, Prop. 4] allows us to obtain an error bound for minimum SW_p transport models.

Proposition 3.13. *Suppose $\eta, \nu \in \mathbb{P}_p(\Omega)$ for $p \geq 1$ and $\mathcal{T} \subseteq L^q_\eta(\Omega;\Omega)$ for $q \geq p$. Let f_θ denote the density of $\pi_\theta \# \nu$ and F_θ denote its corresponding cummulative distribution function (CDF). Assume that the functional $\mathrm{SJ}_p(\nu) := \mathbb{E}_{\theta\sim\varrho} \int_0^1 \left(\frac{\sqrt{t(1-t)}}{f_\theta(F_\theta^{-1}(t))} \right)^p dt < s$ for some $s > 0$. Let $\widehat{T}^N$ be the solution to Eq (3.1) with $D \leftarrow \mathrm{SW}_p$. Then with probability $1 - \delta > 0$ it holds that*

$$\mathrm{SW}_p(\widehat{T}^N \# \eta, \nu) \leq \|\widehat{T}^\dagger - T^\dagger\|_{L^q_\eta(\Omega;\Omega)} + \frac{C}{\delta\sqrt{N}}$$

where $C(s,p) > 0$ is a constant.

3.4.3 FUSE discrepancies

Now, we turn our attention to the FUSE discrepancy introduced in Equation (3.4). We will verify that the properties of the underlying family of discrepanices are inherited by FUSE_γ.

First, we show that FUSE_γ satisfies the near-triangle property, provided that its constituent discrepancies individually satisfy the weak triangle inequality.

Lemma 3.14. *Consider the FUSE_γ discrepancy defined as Equation (3.4) with a measure $\varrho \in \mathbb{P}(I)$ using a family $\{\mathcal{D}_i \mid i \in I\}$, where each D_i satisfies the weak triangle inequality with constant $C_i \geq 0$. Then, for any triple $\mu, \mu'\mu'' \in \mathbb{P}(\Omega)$ and choice of $\gamma > 0$, it holds that*

$$\mathrm{FUSE}_\gamma(\mu,\mu') \leq C\left(\mathrm{FUSE}_{\tau(\gamma)}(\mu',\mu'') + \mathrm{FUSE}_{\tau(\gamma)}(\mu'',\mu')\right).$$

where $C = \operatorname{ess\,sup}_{i\in\operatorname{supp}\varrho} C_i$ and $\tau(\gamma) = 2C\gamma$.

Proof. Using the weak triangle inequality for each D_i and the fact that log-sum-exp is non-decreasing and convex, we have that

$$\begin{aligned}
\mathrm{FUSE}_\gamma(\mu,\mu') &= \frac{1}{\gamma}\log(\mathbb{E}_{i\sim\varrho}\exp[\gamma D_i(\mu,\mu')]) \\
&\leq \frac{1}{\gamma}\log(\mathbb{E}_{i\sim\varrho}\exp[\gamma C_i(D_i(\mu,\mu'') + D_i(\mu'',\mu')]) \\
&\leq \frac{1}{2\gamma}\log(\mathbb{E}_{i\sim\varrho}\exp[2C\gamma(D_i(\mu,\mu'')]) + \frac{1}{2\gamma}\log(\mathbb{E}_{i\sim\varrho}\exp[2C\gamma(D_i(\mu'',\mu')]) \\
&= C(\mathrm{FUSE}_{2C\gamma}(\mu,\mu') + \mathrm{FUSE}_{2C\gamma}(\mu'',\mu')
\end{aligned}$$

□

Next, a straightforward application of Hölder's inequality yields an upper bound on FUSE_γ based on its constituent discrepancies.

Lemma 3.15. *Consider* FUSE_γ *discrepancy defined as Eq (3.4) based on a family* $\{D_i \mid i \in I\}$ *and with a measure* $\varrho \in \mathbb{P}(I)$*. Then it holds that*

$$\mathrm{FUSE}_\gamma(\mu, \mu') \leq \operatorname*{ess\,sup}_{i \in \operatorname{supp} \varrho} D_i(\mu, \mu'),$$

Proof. Using Hölder's inequality and monotonicity of logarithm we can write

$$\mathrm{FUSE}_\gamma(\mu, \mu') \leq \frac{1}{\gamma} \log(\operatorname*{ess\,sup}_{i \in \operatorname{supp} \varrho} \exp(\gamma D_i(\mu, \mu'))) = \operatorname*{ess\,sup}_{i \in \operatorname{supp} \varrho} D_i(\mu, \mu')$$

□

The above lemmas imply that FUSE_γ inherits the desirable properties of its underlying family such as near-triangle property, lipschitz stability, and sample complexities. An application of Theorem 3.5 leads to the following error bound for minimum FUSE transport models. The proof of the following result follows from straightforward applications of the Lemmas Lemmas 3.14 and 3.15 along with the master theorem Theorem 3.5.

Proposition 3.16. *Consider the* FUSE_γ *discrepancy defined using a family* $\{D_i \mid i \in I\}$ *with a measure* $\varrho \in \mathbb{P}(I)$ *that satisfies the conditions of Lemma 3.14. Suppose* $\mathcal{T} \subseteq L^p_\eta(\Omega; \Omega)$ *for some* $p \in [1, +\infty]$*. Assume that the tuples* $(\mathcal{T}, D_i, \eta)$ *are lipschitz stable with constants* $C^i_\eta > 0$ *for all* $i \in \operatorname{supp} \varrho$*. Let* $\widehat{T}^N$ *be the solution to Eq (3.1) with* $D \leftarrow \mathrm{FUSE}_\gamma$*. Then it holds that*

$$\mathrm{FUSE}_{\gamma'}(\widehat{T}^N \# \eta, \nu) \leq C \left[\|\widehat{T}^\dagger - T^\dagger\|_{L^p_\eta(\Omega;\Omega)} + \operatorname*{ess\,sup}_{i \in \operatorname{supp} \varrho} \left(D_i(\nu, \nu^N) + D_i(\nu^N, \nu) \right) \right]$$

where $\gamma' = \frac{\gamma}{\operatorname{ess\,sup}_{i \in \operatorname{supp} \varrho} 2C_i}$ *and* C *depends on* $\operatorname{ess\,sup}_{i \in \operatorname{supp} \varrho} C_i$ *and* $\operatorname{ess\,sup}_{i \in \operatorname{supp} \varrho} C^i_\eta$*.*

MMD-FUSE

The classic example of FUSE_γ type discrepancy is the extension of the MMD that was introduced in [27] for two-sample testing. Consider a family of kernels $\{K_i \mid i \in I\}$ over the index set I and write

MMD_i to denote the MMD arising defined via K_i. Then for distribution $\varrho \in \mathbb{P}(I)$ and parameter $\gamma > 0$, we define the MMD-FUSE$_\gamma$ discrepancy between two measures $\mu, \mu' \in \cap_{i\in I}\mathbb{P}_{K_i}(\Omega)$ as

$$\text{MMD-FUSE}_\gamma(\mu, \mu') = \frac{1}{\gamma}\log\left(\mathbb{E}_{i\sim\varrho}\exp(\gamma\text{MMD}_i^2(\mu, \mu'))\right), \tag{3.9}$$

and infinite otherwise.

Since MMD satisfies the triangle inequality, we can readily verify that squared MMD satisfies the weak triangle inequality with a constant $C = 2$. Moreover, it is symmetric. With additional Lipschitz assumption on the kernels, we can use Proposition 3.16 to get the following error bound for minimum MMD-FUSE transport models.

Proposition 3.17. *Let $\{K_i \mid i \in I\}$ be a family of Mercer kernels satisfying $\|K_i(x,\cdot) - K_i(x',\cdot)\|_K \leq L_i|x - x'|$ for $L_i > 0$ and such that $\operatorname{ess\,sup}_{i\in\text{supp }\varrho}\sup_{x\in\Omega} K_i(x,x) < \infty$ and $\operatorname{ess\,sup}_{i\in\text{supp }\varrho} L_i < \infty$. Let $\widehat{T}^N$ be the solution to Eq (3.1) with $D \leftarrow$ MMD-FUSE$_\gamma$. Then, with probability $1 - \delta > 0$ it holds that*

$$\text{MMD-FUSE}_{\frac{\gamma}{4}}(\widehat{T}\#\eta, \nu) \leq C\left[\|\widehat{T}^\dagger - T^\dagger\|_{L^p_\eta(\Omega;\Omega)} + \frac{16}{N}\left(\operatorname*{ess\,sup}_{i\in\text{supp }\varrho}\sup_{x\in\Omega} K_i(x,x)\right)\right.$$
$$\left. + 8\sqrt{18\log\left(\frac{2}{\delta}\right)} + 18\log\left(\frac{2}{\delta}\right)\right]$$

where C depends on $\operatorname{ess\,sup}_{i\in\text{supp }\varrho} L_i$.

Remark 3.18. *We note that satisfying the assumptions of the above theorem on the kernels K_i is often straightforward in practice. For example, a common choice of the kernel family is a stationary kernel with the index i controlling the lengthscale in which case we can simply normalize the kernels to ensure $\sup_x K_i(x,x) = 1$ for the entire family. Moreover, as long as the lengthscales are bounded away from zero the L_i will also be uniformly bounded. For example, we can take K_I to be a collection of Radial Basis Function (RBF) kernels whose lengthscales σ_i are indexed by I.*

$$K_I = \{K_i | i \in I\}, \quad K_i(x,x') = \exp\left(\frac{-\|x - x'\|^2}{2\sigma_i^2}\right)$$

Sliced Wasserstein-FUSE

We can naturally define a FUSE$_\gamma$ analogue for the Sliced Wasserstein distance over different projections π_θ. For $p \geq 1$, distribution $\varrho \in \mathbb{P}(\mathbb{S}^{d-1})$ and parameter $\gamma > 0$, we define the p-th order

Sliced Wasserstein FUSE discrepancy between two measures $\mu, \mu' \in \mathbb{P}_p(\Omega)$ as

$$\text{SW}_p\text{-FUSE}_\gamma(\mu, \mu') = \left(\frac{1}{\gamma} \log \left(\mathbb{E}_{\theta \sim \varrho} \exp(\gamma W_p^p(\pi_\theta \# \mu, \pi_\theta \# \mu')) \right) \right)^{1/p}, \tag{3.10}$$

Note that we recover the standard Sliced Wasserstein distance when $\gamma \to 0$. On the other hand, the $\gamma \to \infty$ limit corresponds to the Max Wasserstein Distance that has attracted significant interest in both optimal transport and generative modeling literature [150, 61, 168].

We can readily verify that the W_p^p satisfies the weak triangle inequality with the constant $C = 2^{p-1}$. This follows from the fact that $|x+y|^p \leq 2^{p-1}(|f|^p + |g|^p)$ due to the convexity of $|\cdot|^{1/p}$ for $p \geq 1$. From Section 3.4.2, we know that $(\pi_\theta \circ L_\eta^q(\Omega;\Omega), W_p^p, \eta)$ is Lipschitz stable with constant $C = 1$ for any $\eta \in \mathbb{P}_p(\Omega)$ and $q \geq p$. Moreover, it is also symmetric. For one dimensional measures, [31] derive the empirical convergence rates for W_p^p provided that the p-th moment of the measure is bounded. Then, a straightforward application of Proposition 3.16 and empirical convergence rates for $W_p^p(\pi_\theta \# \nu, \pi_\theta \# \nu^N)$ using [31, Thm 5.3] gives us the following error bound for minimum SW_p-FUSE estimators.

Proposition 3.19. *Suppose $\eta, \nu \in \mathbb{P}_p(\Omega)$ for $p \geq 1$ and $\mathcal{T} \subseteq L_\eta^q(\Omega;\Omega)$ for $q \geq p$. Let f_θ denote the density of $\pi_\theta \# \nu$ and F_θ denote its corresponding CDF. Define the functional $\text{J}_{p,\theta}(\nu) := \int_0^1 \left(\frac{\sqrt{t(1-t)}}{f_\theta(F_\theta^{-1}(t))} \right)^p dt$. Assume that $\operatorname{ess\,sup}_{\theta \in \text{supp } \varrho} \text{J}_{p,\theta}(\nu) < s$ for some $s > 0$. Let $\widehat{T}^N$ be the solution to Eq (3.1) with $D \leftarrow \text{SW}_p\text{-FUSE}_\gamma$. Then with probability $1 - \delta > 0$ it holds that*

$$\text{SW}_p\text{-FUSE}_{\gamma/2^p} \leq C \left[\|\widehat{T}^\dagger - T^\dagger\|_{L_\eta^q(\Omega;\Omega)} + \frac{C}{\delta} \cdot (N+2)^{-p/2} \right]$$

where the constant $C(p,s) > 0$

3.4.4 Controlling approximation errors

So far in our analysis, we haven't made any assumptions on the true map $T^\dagger$ except that it belongs to $\mathcal{T}$, which we take to be $L_\eta^p(\Omega;\Omega)$. In order to get provable rate of convergence for the term $\|\widehat{T}^\dagger - T^\dagger\|_{\mathcal{T}}$ which appears in all of the bounds, we will make additional regularity assumptions on $T^\dagger$ such as various levels of Sobolev or Hölder smoothness, that will allow us to achieve quantitative rates for this term.

In particular, we will assume that $T^\dagger \in \mathcal{H}^t_\eta(\Omega;\Omega)$. Consider the multi-index $\mathbf{j} = (j_1, j_2, \ldots, j_d)$ and define the mixed derivative $\partial^{|\mathbf{j}|_1} f(x) := \frac{\partial^{|\mathbf{j}|_1} f}{\partial^{j_1}_{x_1} \ldots \partial^{j_d}_{x_d}}(x)$. Then, we define the Sobolev space $H^t_\eta(\Omega;\Omega)$ to be the space of functions $f : \Omega \to \Omega$ with mixed derivatives of degree $\leq k$ in $L^2_\eta(\Omega;\Omega)$ equipped with the norm $\|f\|_{H^t_\eta(\Omega;\Omega)} = \left(\sum_{\|\mathbf{j}\|_1 \leq t} \|\partial^{|\mathbf{j}|_1} f\|^2_{L^2_\eta(\Omega;\Omega)}\right)^{1/2}$. That is, $H^t_\eta(\Omega;\Omega) = \{f \in L^2_\eta(\Omega;\Omega) \mid \|f\|_{H^t(\Omega;\Omega)} < +\infty\}$. Now, we will show error bounds for specific choice of transport maps.

Polynomials

Let our parametric model class $\mathcal{S}$ be the space of polynomials of degree α. Suppose $\mathbf{m} = (m_1, \ldots, m_d)$ be a multi-index with non-negative integers m_i. Let $p_{\mathbf{m}} := \Pi^d_{i=1} p_{m_i}(x_i)$ be a multi-dimensional polynomial of degree $\mathbf{m}$ where each co-ordinate of $p_{m_i} : \mathbb{R} \to \mathbb{R}$ is the corresponding univariate polynomial of degree m_i for each $1 \leq i \leq d$. We then take

$$\mathcal{S} := \{T \in L^2_\eta(\Omega;\Omega) \mid T_i(x) = \sum_{|\mathbf{m}|_1 \leq \alpha} c^i_{\mathbf{m}} p_{\mathbf{m}}(x), \quad i = 1, \ldots d\}. \tag{3.11}$$

If the domain is rectangular, we take $p_{m_i}(x)$ to be the Legendre polynomial of degree m_i. For any $x \in [-1, 1]$, a legendre polynonial of degree α is defined as

$$p_\alpha(x) = \frac{1}{2^\alpha \alpha!} \frac{d^\alpha}{dx^\alpha} \left((x^2 - 1)^n\right).$$

When the domain is unbounded, we take $p_{m_i}(x)$ to be the Hermite polynomial of degree m_i defined as

$$p_\alpha(x) = (-1)^\alpha \exp(x^2) \frac{d^\alpha}{d_{x^\alpha}} \exp(-x^2).$$

We recall the following classic error bounds for projections of Sobolev functions onto the span of Hermite and Legendre polynomials from [48, Thm 2.3] and [41, Thm 2.1] respectively.

Proposition 3.20. *Let $\mathcal{T} = L^2_\eta(\Omega;\Omega)$ and $T^\dagger \in H^t_\eta(\Omega;\Omega) \subseteq \mathcal{T}$. Recall our notation that $\widehat{T}^\dagger$ is the best $L^2(\Omega)$-approximation of $T^\dagger$ in the class $\mathcal{S}$.*

1. *(Unbounded domain) Let $\Omega = \mathbb{R}^d$ and η is Gaussian. Take $\mathcal{S}$ to be the span of Hermite polynomials of degree α. Then, with constant $C(\Omega) > 0$, it holds that*

$$\|\widehat{T}^\dagger - T^\dagger\|_{L^2_\eta(\Omega;\Omega)} \leq C\alpha^{-t/2} \|T^\dagger\|_{H^t_\eta(\Omega;\Omega)}$$

2. *(Bounded domain) Let* $\Omega = [-1,1]^d$ *and* η *is the uniform measure. Take* $\mathcal{S}$ *to be the span of Legendre polynomials of degree* α*. Then, with constant* $C(\Omega) > 0$*, it holds that*

$$\|\widehat{T}^\dagger - T^\dagger\|_{L^2_\eta(\Omega;\Omega)} \leq C\alpha^{-(t+\frac{1}{2})}\|T^\dagger\|_{\mathcal{H}^t_\eta(\Omega;\Omega)}$$

Combining this with Theorem 3.5, we can derive the error bounds for polynomial transport maps minimizing specific discrepancies. As an example, we show an error bound for Hermite polynomial transport maps minimizing the W_2 distance using Proposition 3.12 and Proposition 3.20(1).

Proposition 3.21. *Suppose* $\Omega = \mathbb{R}^d$*,* $\eta, \nu \in \mathbb{P}_2(\Omega)$ *and* η *be the Gaussian measure. Let* $\mathcal{T} \subseteq L^2_\eta(\Omega;\Omega)$ *and assume there exists* $T^\dagger \in H^t_\eta(\Omega;\Omega)$ *such that* $T^\dagger \# \eta = \nu$*. Let* $\mathcal{S}$ *be the set of transport maps with Hermite polynomial basis. Let* $\widehat{T}^N$ *be the solution to Eq (3.1) over* $\mathcal{S}$ *with* $D \leftarrow W_2$*. Then with probability* $1 - \delta > 0$ *it holds that*

$$W_2(\widehat{T}^N \# \eta, \nu) \leq C \left[\alpha^{-t/2}\|T^\dagger\|_{H^t_\eta(\Omega;\Omega)} + \frac{1}{\delta} \cdot \begin{cases} N^{-1/4} & d < 4, \\ N^{-1/4}\log(1+N)^{1/2} & d = 4, \\ N^{-1/d} & d > 4, \end{cases} \right]$$

where $C(d, \Omega, \nu) > 0$ *is a constant.*

Remark For specific conditions on ν, we have results on the smoothness properties of the W_2 optimal map $T^\dagger$. If ν has unbounded support and is strongly log-concave, then $T^\dagger$ lies in the weighted Sobolev space $H^2_\eta(\Omega;\Omega)$ [14, Sec 4.2.2]. If the density of ν is $C^t(\Omega)$ and ν has convex and compact support, then $T^\dagger$ lies in $\mathcal{H}^{t+1}_\eta(\Omega;\Omega)$ [14, Sec 4.2.1].

RKHS

Now, we consider our parametric model class to be a subset of an RKHS $(\mathcal{H}_K, \|\cdot\|_K)$ with the reproducing kernel K. Let $X = \{x_1, \ldots, X_M\} \subset \Omega$ be a collection of distinct points with fill distance $h_X := \sup_{x\in\Omega}\|x - x'\|_2$. Then, we take our model class $\mathcal{S}$ to be the span of kernels centered on the points in X.

The following result is a straightforward extension of classic Sobolev sampling inequalities, see for example [228, Corollary 11.33].

Proposition 3.22. *Let Ω be a bounded open set with Lipschitz boundary and let η be the uniform measure on Ω. Suppose $(\mathcal{H}_K, \|\cdot\|_K)$ is an RKHS that is continuously embedded in $H^t(\Omega)$ with $t > d/2$ and $T^\dagger \in \mathcal{H}_K$. Let $X = \{x_1, \ldots, X_M\} \subset \Omega$ be a set of distinct points as above with fill distance $h_X := \sup_{x\in\Omega} \|x - x'\|_2$ and take $\mathcal{S} = span(K(x, x_i), x_i \in X)$. Finally, let $\widehat{T}^\dagger$ be the best $L^2(\Omega)$-approximation of $T^\dagger$ in $\mathcal{S}$. Then, there exists $h_0 > 0$ such that for $h_X \leq h_0$ it holds that*

$$\|\widehat{T}^\dagger - T^\dagger\|_{L^2_\eta(\Omega;\Omega)} \leq C h_X^t \|T^\dagger\|_K$$

where $C(\Omega) > 0$.

Combining this with Theorem 3.5, we can derive the error bounds for RKHS transport maps minimizing specific discrepancies. As an example application, we derive an error bound for RKHS transport maps minimizing MMD using Corollary 3.10 and Proposition 3.22.

Corollary 3.23. *Let Ω be a bounded open set with Lipschitz boundary and η be the uniform measure on Ω. Let $K : \Omega \times \Omega \to \mathbb{R}$ be a Mercer kernel satisfying $\|K(x,\cdot) - K(x',\cdot)\|_K \leq L|x - x'|$ for a constant $L > 0$, $\mathcal{T} = L^2_\eta(\Omega;\Omega)$ and $\nu \in \mathbb{P}_K(\Omega)$. Let $V : \Omega \times \Omega \to \mathbb{R}^d$ be a vector-valued Mercer Kernel with the corresponding RKHS $(\mathcal{H}_V, \|.\|_V)$ that is continuously embedded in $H^t(\Omega)$ for $t > d/2$. Assume $T^\dagger \in \mathcal{H}_V$. Take $\mathcal{S}$ to be the span of kernel V centered on a set of distinct points X as defined above. Let $\widehat{T}^N$ be the solution to Eq (3.1) with $D \leftarrow \mathrm{MMD}_K$. Then with probability $1 - \delta > 0$ it holds that*

$$\mathrm{MMD}_K(\widehat{T}^N \# \eta, \nu) \leq C h_X^t \|T^\dagger\|_K + \sqrt{\frac{1}{N} \sup_{x\in\Omega} K(x,x)} \left(4 + \sqrt{18 \log\left(\frac{2}{\delta}\right)}\right).$$

where $C(L, \Omega) > 0$.

3.5 Numerical experiments

In this section, we empirically investigate the generalization bounds derived in Theorem 3.5 for several parametric models classes $\mathcal{S}$ and discrepancies D. The focus is to validate the error decomposition into approximation and stochastic variance terms, assessing whether the observed rates align with the theoretical predictions. We examine two key regimes: (1) the stochastic error-dominated regime where the error scales as a function of the number of samples N, and (2) the approximation error-dominated regime, where the error depends on the properties of function class $\mathcal{S}$.

3.5.1 Minimum MMD RKHS transport maps

We first validate the bound in Corollary 3.23 for transport maps in an RKHS. Specifically, we consider the case where $\Omega \subseteq \mathbb{R}^2$ and the kernel K is the RBF kernel. For transport maps, we use vector-valued Matérn kernels V with smoothness parameters $\rho = 0.5, 1.5$, and 2.5. From [218, Cor. A.6], the RKHS induced by a Matérn kernel is equivalent to the Sobolev space $H^{\rho+d/2}$ for $\rho + d/2 > d/2$, and we prescribe $T^\dagger \in \mathcal{H}_V$. We take η to be Gaussian and generate N samples. Then, we generate samples from $T^\dagger \# \eta$.

The generalization error bound in Corollary 3.23 predicts:

$$\mathrm{MMD}_K(\widehat{T}^N \# \eta, \nu) \leq C h_X^{\rho+1} \|T^\dagger\|_K + \sqrt{\frac{1}{N} \sup_{x \in \Omega} K(x,x)} \left(4 + \sqrt{18 \log\left(\frac{2}{\delta}\right)}\right),$$

Here, h_X which represents the fill distance of the set X scales as $O(M^{-1/d})$ with $M = |X|$ denoting the number of kernel centers and d is the dimension [182].

First, we analyze the regime when $M \gg 1$ (ensuring $h_X \ll 1$) and the stochastic term dominates the generalization error. In this regime, we expect the error to scale as $O(1/\sqrt{N})$, independent of the kernel smoothness. The experimental results, shown in Figure 3.1 (left), confirm this behavior, with the variance decreasing empirically rate of $O(N^{-0.41})$. Moreover, we see this regime only when M is sufficiently large.

When $N \gg 1$, the generalization error is governed by the approximation term $O(h_X^{\rho+1})$, which depends on the kernel smoothness ρ and the fill distance h_X. Interestingly, we observe faster-than-expected decay rates in Figure 3.1 (right). This suggests that the theoretical bound may be loose or that the method is approximating another map that is smoother but maintains comparable generalization error as the true $T^\dagger$.

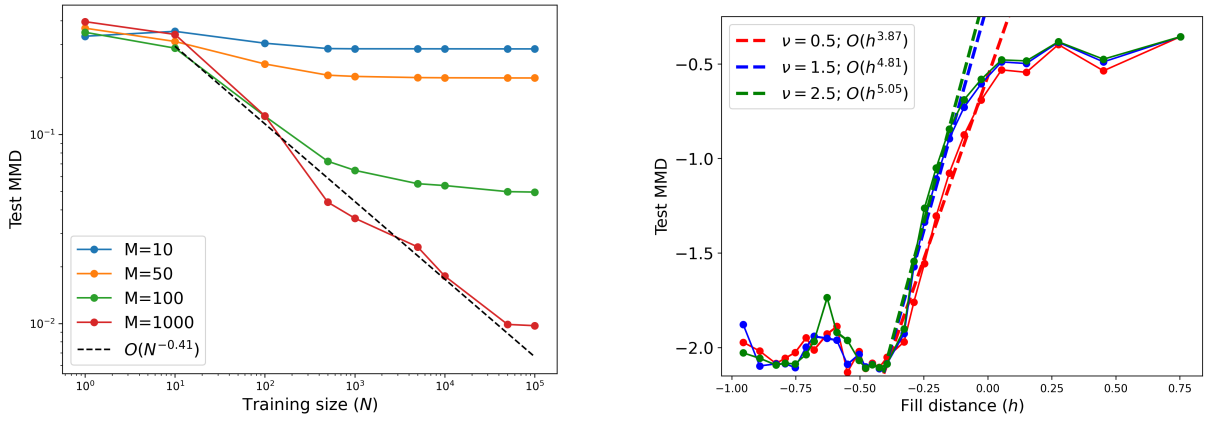

Figure 3.1: Error analysis for RKHS transport maps that minimize MMD (Left) Stochastic variance error. (Right) Approximation error.

Chapter 4

STRUCTURED RANDOM RECEPTIVE FIELDS ENABLE INFORMATIVE SENSORY ENCODINGS

4.1 Prelude to the article

In the quest to understand how learning systems extract meaningful information from vast amounts of data, both neuroscience and artificial intelligence communities have converged on the importance of representations. The ability to focus on critical aspects of the environment while ignoring irrelevant details is a hallmark of intelligent behavior. In biological systems, sensory neurons transform raw stimuli into actionable information by detecting specific features and patterns. For instance, early visual cortical neurons are sensitive to orientations, spatial frequencies, and luminance, enabling organisms to interpret complex visual environments [107]. Similarly, in artificial systems, effective representation learning is crucial for building models that generalize well and perform efficiently [22].

This chapter explores the interplay between biological sensory representations and artificial learning algorithms, emphasizing how insights from both fields can inform and enhance each other. Neuroscience provides detailed models of how sensory systems extract and process information, which can inspire new computational frameworks in machine learning. Conversely, machine learning offers theoretical tools and models that help analyze neuronal learning systems, shedding light on the underlying computational principles of the brain.

Traditional models of sensory processing, such as the linear-nonlinear (LN) model [49], conceptualize neurons as linear filters followed by a nonlinear activation function. While these models have been successful in capturing certain aspects of neuronal responses, they often fall short in explaining the full complexity observed in naturalistic settings [43]. Moreover, they may not account for higher-order effects such as context-dependent modulation and feedback mechanisms [71].

In parallel, the machine learning community has made significant strides in representation

learning through the development of deep neuronal networks and kernel methods. Neural networks, especially deep learning models, automatically learn hierarchical representations of data by optimizing large number of parameters through backpropagation [22]. Kernel methods, on the other hand, implicitly map inputs into high-dimensional feature space where linear separation becomes feasible [197]. Both approaches have their advantages but also limitations, such as interpretability challenges in deep networks and the necessity of choosing the right high-dimensional space via the choice of the kernel.

A promising avenue lies in the cross-pollination between neuroscience and machine learning, where insights from biological systems inspire new computational models, and artificial systems provide frameworks and theories for testing and explaining hypotheses about brain function [101]. For example, convolutional neural networks (CNNs) have architecture reminiscent of the hierarchical processing in the visual cortex and have been used to model neural responses in both the ventral visual stream [232] and auditory cortex [119]. At the same time, principles from neuroscience have inspired architectural innovations in artificial networks, leading to improvements in robustness and efficiency [57].

In this work, we contribute to the interdisciplinary dialogue by developing a framework that incorporates sensory representations into random feature networks (RFNs). Specifically, we model the receptive fields of sensory neurons as random samples from Gaussian Processes (GPs) with structured covariance functions. This approach captures the variability and structure observed in biological receptive fields while maintaining mathematical tractability and interpretability.

Our theoretical analysis in Section 4.3 reveals that networks with such structured random weights perform a randomized transformation and filtering of inputs, effectively projecting them into a basis that emphasizes relevant features while suppressing noise. This insight aligns with the concept of inductive bias in machine learning, where prior knowledge about the structure of data is used to guide learning algorithms towards better generalization [22].

Empirically, we demonstrate in Section 4.3.3 and Section 4.3.4 that this framework benefits both fields. For neuroscience, it provides a generative model that matches experimental data from mechanosensory neurons in the gyroscopic organs of insects [77] and simple cells in the mammalian primary visual cortex (V1) [116]. The generated receptive fields, their second-order statistics, and

their principal components closely resemble those observed in biological systems. For machine learning, incorporating these biologically inspired receptive fields into RFNs leads to more efficient learning on synthetic tasks, requiring fewer neurons and training examples, and achieving faster convergence.

Our work underscores the mutual benefits of integrating biological principles into artificial learning systems and using machine learning theory to analyze neuronal systems. By bridging the gap between neuroscience and machine learning, we enhance our understanding of sensory representation and offer practical advantages of building more efficient and robust artificial networks. This interdisciplinary approach opens avenues for future research, such as exploring other sensory modalities, incorporating temporal dynamics and feedback mechanisms, and extending the framework to deeper network architectures.

The contents of this chapter appear in the article [160] that was published in the journal PLOS Computational Biology (PLOS CB).

4.2 Introduction

It has long been argued that the brain uses a large population of neurons to represent the world [237, 79, 194, 208]. In this view, sensory stimuli are encoded by the responses of the population, which are then used by downstream areas for diverse tasks, including learning, decision-making, and movement control. These sensory areas have different neurons responding to differing stimuli while also providing a measure of redundancy. However, we still lack a clear understanding of what response properties are well-suited for different sensory modalities.

One way to approach sensory encoding is by understanding how a neuron would respond to arbitrary stimuli. Experimentally, we typically present many stimuli to the animal, measure the responses of sensory neurons, then attempt to estimate some kind of model for how the neurons respond to an arbitrary stimulus. A common assumption is that the neuron computes a linear filter of the stimulus, which then drives spiking through a nonlinear spike-generating mechanism. Mathematically, this assumption can be summarized as the number of measured spikes for a stimulus $\mathbf{x}$ being equal to $\sigma(\mathbf{w}^T\mathbf{x})$ for a weight vector $\mathbf{w}$ and nonlinearity σ. Here, the weights $\mathbf{w}$ define the filtering properties of the neuron, also known as its *receptive field* [198]. This model is known as a

linear-nonlinear (LN) model [49], and it is also the most common form of artificial neuron in artificial neural networks (ANNs). LN models have been used extensively to describe the firing of diverse neurons in various sensory modalities of vertebrates and invertebrates. In the mammalian visual system, LN models have been used to characterize retinal ganglion cells [192], lateral geniculate neurons [52], and simple cells in primary visual cortex (V1) [116]. They have also been used to characterize auditory sensory neurons in the avian midbrain [121] and somatosensory neurons in the cortex [191]. In insects, they have been used to understand the response properties of visual interneurons [188], mechanosensory neurons involved in proprioception [77, 173], and auditory neurons during courtship behavior [53].

Given the stimulus presented and neural response data, one can then estimate the receptive fields of a population of neurons. Simple visual receptive fields have classically been understood as similar to wavelets with particular spatial frequency and angular selectivity [116]. In mechanosensory areas, receptive fields are selective to temporal frequency over a short time window [77]. Commonly, parametric modeling (Gabor wavelets [208]) or smoothing (regularization, etc. [166]) are used to produce "clean" receptive fields. Yet, the data alone show noisy receptive fields that are perhaps best modeled using a random distribution [32]. As we will show, modeling receptive fields as random samples produces realistic receptive fields that reflect both the structure and noisiness seen in experimental data. More importantly, this perspective creates significant theoretical connections between foundational ideas from neuroscience and artificial intelligence. This connection helps us understand why receptive fields have the structures that they do and how this structure relates to the kinds of stimuli that are relevant to the animal.

Modeling the filtering properties of a population of LN neurons as samples from a random distribution leads to the study of networks with random weights [187, 44, 133]. In machine learning (ML), such networks are known as *random feature networks* (RFNs) [38, 109, 177, 134]. The study of RFNs has rapidly gained popularity in recent years, in large part because it offers a theoretically tractable way to study the learning properties of ANNs where the weights are tuned using data [9, 8, 46]. When the RFN contains many neurons, it can approximate functions that live in a well-understood function space. This function space is called a *reproducing kernel Hilbert space* (RKHS), and it depends on the network details, in particular the weight i.e. receptive field

distribution [148, 229, 178]. Learning can then be framed as approximating functions in this space from limited data.

Several recent works highlight the RFN theory's usefulness for understanding learning in neural systems. Bordelon, Canatar, and Pehlevan, in a series of papers, have shown that neural codes allow learning from few examples when spectral properties of their second-order statistics aligns with the spectral properties of the task [35, 36, 40]. When applied to V1, they found that the neural code is aligned with tasks that depend on low spatial frequency components. Harris constructed an RFN model of sparse networks found in associative centers like the cerebellum and insect mushroom body and showed that these areas may behave like additive kernels [99], an architecture also considered by Hashemi et al. [100]. These classes of kernels are beneficial for learning in high dimensions because they can learn from fewer examples and remain resilient to input noise or adversarial perturbation. Xie et al. investigated the relationship between the fraction of active neurons in a model of the cerebellum—controlled by neuron thresholds—and generalization performance for learning movement trajectories [230]. In the vast majority of network studies with random weights, these weights $\mathbf{w}$ are drawn from a Gaussian distribution with independent entries. This sampling is equivalent to a fully *unstructured* receptive field, which looks like white noise.

Closely related to our work, a previous study of ANNs showed that directly learning structured receptive fields could improve image classification in deep networks [113]. Their receptive fields were parametrized as a sum of Gaussian derivatives up to fourth order. This led to better performance against rival architectures in low data regimes.

In this article, we study the effect of having *structured yet random* receptive fields and how they lead to informative sensory encodings. Specifically, we consider receptive fields generated by a Gaussian process (GP), which can be thought of as drawing the weights $\mathbf{w}$ from a Gaussian distribution with a particular covariance matrix. We show that networks with such random weights project the input to a new basis and filter out particular components. This theory introduces realistic structure of receptive fields into random feature models which are crucial to our current understanding of artificial networks. Next, we show that receptive field datasets from two disparate sensory systems, mechanosensory neurons on insect wings and V1 cortical neurons from mice and monkeys, are well-modeled by GPs with covariance functions that have wavelet eigenbases. Given

the success of modeling these data with the GP, we apply these weight distributions in RFNs that are used in synthetic learning tasks. We find that these structured weights improve learning by reducing the number of training examples and the size of the network needed to learn the task. Thus, structured random weights offer a realistic generative model of the receptive fields in multiple sensory areas, which we understand as performing a random change of basis. This change of basis enables the network to represent the most important properties of the stimulus, which we demonstrate to be useful for learning.

4.3 Results

We construct a generative model for the receptive fields of sensory neurons and use it for the weights of an ANN. We refer to such a network as a *structured* random feature network. We first review the basics of random feature networks, the details and rationale behind our generative model, and the process by which we generate hidden weights. Our main theory result is that networks with such weights transform the inputs into a new basis and filter out particular components, thus bridging sensory neuroscience and the theory of neural networks. Next, we show that neurons in two receptive field datasets—insect mechanosensory neurons and mammalian V1 cortical neurons—are well-described by our generative model. There is a close resemblance between the the second-order statistics, sampled receptive fields, and their principal components for both data and model. Finally, we show the performance of structured random feature networks on several synthetic learning tasks. The hidden weights from our generative model allows the network to learn from fewer training examples and smaller network sizes.

Theoretical analysis

We consider receptive fields generated by GPs in order to connect this foundational concept from sensory neuroscience to the theory of random features in artificial neural networks. GPs can be thought of as samples from a Gaussian distribution with a particular covariance matrix, and we initialize the hidden weights of RFNs using these GPs. We show that using a GP causes the network to project the input into a new basis and filter out particular components. The basis itself is determined by the covariance matrix of the Gaussian, and is useful for removing irrelevant and

noisy components from the input. We use these results to study the space of functions that RFNs containing many neurons can learn by connecting our construction to the theory of kernel methods.

Random feature networks

We start by introducing the main learning algorithm and the neuronal model of our work, the RFN. Consider a two-layer, feedforward ANN. Traditionally, all the weights are initialized randomly and learned through backpropagation by minimizing some loss objective. In sharp contrast, RFNs have their hidden layer weights sampled randomly from some distribution and fixed. Each hidden unit computes a random feature of the input, and only the output layer weights are trained (Figure 4.1). Note that the weights are randomly drawn but the neuron's response is a deterministic function of the input given the weights.

Mathematically, we have the hidden layer activations and output given by

$$\mathbf{h} = \sigma\left(\mathbf{W}\mathbf{x}\right), \qquad \hat{y} = \beta^T \mathbf{h} + \beta_0, \tag{4.1}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the stimulus, $\mathbf{h} = [h_1, h_2, \ldots, h_m]^T \in \mathbb{R}^m$ are the hidden neuron responses, and $\hat{y} \in \mathbb{R}$ is the predicted output. We use a rectified linear (ReLU) nonlinearity, $\sigma(x) = \max(0, x)$ applied entrywise in Eq (4.1). The hidden layer weights $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m]^T \in \mathbb{R}^{m \times d}$ are drawn randomly and fixed. Only the readout weights β_0 and β are trained in RFNs.

In our RFN experiments, we train the readout weights $\beta \in \mathbb{R}^m$ and offset $\beta_0 \in \mathbb{R}$ using a support vector machine (SVM) classifier with squared hinge loss and ℓ^2 penalty with regularization strength of tuned in the range $[10^{-3}, 10^3]$ by 5-fold cross-validation. Our RFNs do not include a threshold for the hidden neurons, although this could help in certain contexts [230].

In the vast majority of studies with RFNs, each neuron's weights $\mathbf{w} \in \mathbb{R}^d$ are initialized i.i.d. from a spherical Gaussian distribution $\mathbf{w} \sim \mathcal{N}(0, \mathbf{I}_d)$. We will call networks built this way *classical unstructured* RFNs (Fig 4.1). We propose a variation where hidden weights are initialized $\mathbf{w} \sim \mathcal{N}(0, \mathbf{C})$, where $\mathbf{C} \in \mathbb{R}^{d \times d}$ is a positive semidefinite covariance matrix. We call such networks *structured* RFNs (Fig 4.1), to mean that the weights are random with a specified covariance. To compare unstructured and structured weights on equal footing, we normalize the covariance matrices so that $\mathrm{Tr}(\mathbf{C}) = \mathrm{Tr}(\mathbf{I}_d) = d$, which ensures that the mean square amplitude of the weights $\mathbb{E}[\|\mathbf{w}\|^2] = d$.

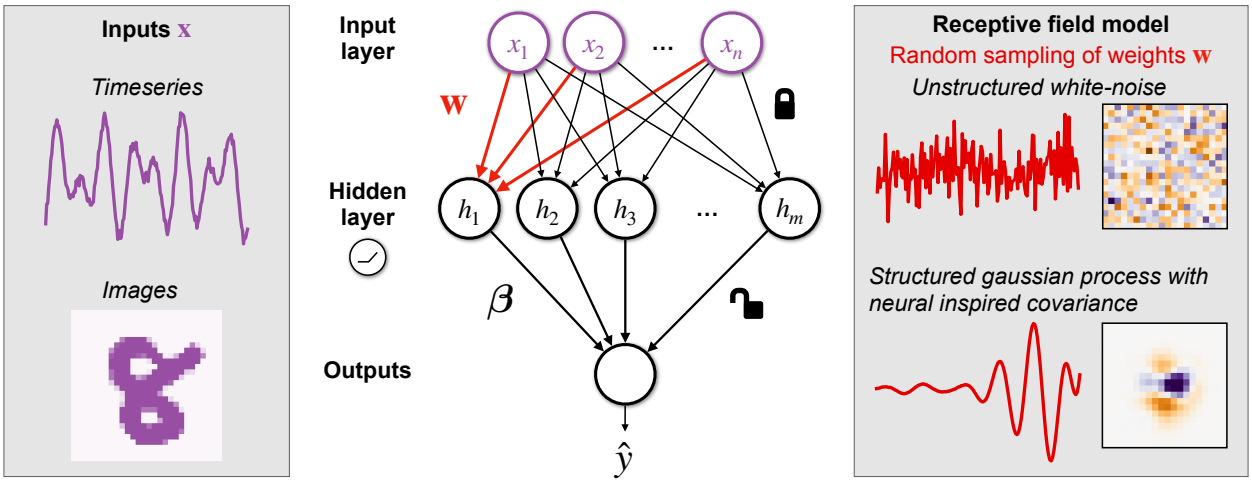

Figure 4.1: **Random feature networks with structured weights.** We study random feature networks as models for learning in sensory regions. In these networks, each neuron's weight $\mathbf{w}$ is fixed as a random sample from some specified distribution. Only the readout weights β are trained. In particular, we specify distributions to be Gaussian Processes (GPs) whose covariances are inspired by biological neurons; thus, each realization of the GP resembles a biological receptive field. We build GP models of two sensory areas that specialize in processing timeseries and image inputs. We initialize $\mathbf{w}$ from these *structured* GPs and compare them against initialization from *unstructured* white-noise distribution.

4.3.1 Receptive fields modeled by linear weights

Sensory neurons respond preferentially to specific features of their inputs. This stimulus selectivity is often summarized as a neuron's receptive field, which describes how features of how the sensory space elicits responses when stimulated [198]. Mathematically, receptive fields are modeled as a linear filter in the stimulus space. Linear filters are also an integral component of the widely used LN model of sensory processing [49]. According to this model, the firing rate of a neuron is a nonlinear function applied to the projection of the stimulus onto the low-dimensional subspace of the linear filter.

A linear filter model of receptive fields can explain responses of individual neurons to diverse stimuli. It has been used to describe disparate sensory systems like visual, auditory, and somatosensory systems of diverse species including birds, mammals, and insects [192, 188, 121, 53, 191]. If the stimuli are uncorrelated, the filters can be estimated by computing the spike triggered average (STA), the average stimulus that elicited a spike for the neuron. When the stimuli are correlated, the STA filter is whitened by the inverse of the stimulus covariance matrix [161]. Often these STAs are denoised by fitting a parametric function to the STA [49], such as Gabor wavelets for simple cells in V1 [116].

We model the receptive field of a neuron i as its weight vector $\mathbf{w}_i$ and its nonlinear function as σ. Instead of fitting a parametric function, we construct covariance functions so that each realization of the resulting Gaussian process resembles a biological receptive field (Fig 4.1).

4.3.2 Structured weights project and filter input into the covariance eigenbasis

We generate network weights from Gaussian processes (GP) whose covariance functions are inspired by the receptive fields of sensory neurons in the brain. By definition, a GP is a stochastic process where finite observations follow a Gaussian distribution [179]. We find that networks with such weights project inputs into a new basis and filter out irrelevant components. We will see that this adds an inductive bias to classical RFNs for tasks with naturalistic inputs and improves learning.

We view our weight vector $\mathbf{w}$ as the finite-dimensional discretization of a continuous function $w(t)$ which is a sample from a GP. The continuous function has domain T, a compact subset of $\mathbb{R}^D$, and we assume that T is discretized using a grid of d equally spaced points $\{t_1, \ldots, t_d\} \subset T$, so that $w_i = w(t_i)$. Let the input be a real-valued function $x(t)$ over the same domain T, which could represent a finite timeseries ($D = 1$), an image of luminance on the retina ($D = 2$), or more complicated spatiotemporal sets like a movie ($D = 3$). In the continuous setting, the d-dimensional ℓ^2 inner product $\mathbf{w}^T\mathbf{x} = \sum_{i=1}^{d} w_i x_i$ gets replaced by the $L^2(T)$ inner product $\langle w, x \rangle = \int_{t \in T} w(t)x(t)\mathrm{d}t$.

Every GP is fully specified by its mean and covariance function $C(t, t')$. We will always assume that the mean is zero and study different covariance functions. By the Kosambi-Karhunen–Loève

theorem [125], each realization of a zero-mean GP has a random series representation

$$w(t) = \sum_{i=1}^{\infty} z_i \lambda_i \phi_i(t), \tag{4.2}$$

in terms of standard Gaussian random variables $z_i \sim \mathcal{N}(0,1)$, functions $\phi_i(t)$, and weights $\lambda_i \geq 0$. The pairs (λ_i^2, ϕ_i) are eigenvalue, eigenfunction pairs of the covariance operator $\mathcal{C} : L^2(T) \to L^2(T)$,

$$(\mathcal{C}f)(t) = \int_{t \in T} C(t,t') f(t') \mathrm{d}t',$$

which is the continuous analog of the covariance matrix $\mathbf{C}$. If $C(t,t')$ is positive definite, as opposed to just semidefinite, all $\lambda_i^2 > 0$ and these eigenfunctions ϕ_i form a complete basis for $L^2(T)$. Using Eq (4.2), the inner product between a stimulus and a neuron's weights is

$$\langle w, x \rangle = \left\langle \sum_{i=1}^{\infty} z_i \lambda_i \phi_i, x \right\rangle = \sum_{i=1}^{\infty} z_i \lambda_i \langle \phi_i, x \rangle = \sum_{i=1}^{\infty} z_i \tilde{x}_i, \quad \text{where } \tilde{x}_i = \lambda_i \langle \phi_i, x \rangle. \tag{4.3}$$

Eq (4.3) shows that the structured weights compute a *projection* of the input x onto each eigenfunction $\langle \phi_i, x \rangle$ and reweight or *filter* by the eigenvalue λ_i before taking the ℓ^2 inner product with the random Gaussian weights z_i.

It is illuminating to see what these continuous equations look like in the d-dimensional discrete setting. Samples from the finite-dimensional GP are used as the hidden weights in RFNs, $\mathbf{w} \sim \mathcal{N}(0, \mathbf{C})$. First, the GP series representation Eq (4.2) becomes $\mathbf{w} = \mathbf{\Phi \Lambda z}$, where $\mathbf{\Lambda}$ and $\mathbf{\Phi}$ are matrices of eigenvalues and eigenvectors, and $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}_d)$ is a Gaussian random vector. By the definition of the covariance matrix, $\mathbf{C} = \mathbb{E}[\mathbf{w}\mathbf{w}^T]$, which is equal to $\mathbf{\Phi \Lambda}^2 \mathbf{\Phi}^T$ after a few steps of linear algebra. Finally, Eq (4.3) is analogous to $\mathbf{w}^T \mathbf{x} = \mathbf{z}^T \mathbf{\Lambda \Phi}^T \mathbf{x}$. Since $\mathbf{\Phi}$ is an orthogonal matrix, $\mathbf{\Phi}^T \mathbf{x}$ is equivalent to a change of basis, and the diagonal matrix $\mathbf{\Lambda}$ shrinks or expands certain directions to perform filtering. This can be summarized in the following theorem:

Theorem 4.1 (Basis change formula)**.** *Assume* $\mathbf{w} \sim \mathcal{N}(0, \mathbf{C})$ *with* $\mathbf{C} = \mathbf{\Phi \Lambda}^2 \mathbf{\Phi}^T$ *its eigenvalue decomposition. For* $\mathbf{x} \in \mathbb{R}^d$*, define*

$$\tilde{\mathbf{x}} := \mathbf{\Lambda \Phi}^T \mathbf{x}. \tag{4.4}$$

Then $\mathbf{w}^T \mathbf{x} = \mathbf{z}^T \tilde{\mathbf{x}}$ *for* $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}_d)$.

Theorem 4.1 says that projecting an input onto a structured weight vector is the same as first filtering that input in the GP eigenbasis and doing a random projection onto a spherical random

Gaussian. The form of the GP eigenbasis is determined by the choice of the covariance function. If the covariance function is compatible with the input structure, the hidden weights filter out any irrelevant features or noise in the stimuli while amplifying the descriptive features. This inductive bias facilitates inference on the stimuli by any downstream predictor. Because the spherical Gaussian distribution is the canonical choice for unstructured RFNs, there is a simple way to evaluate the effective kernel of structured RFNs as $k_{\text{struct}}(\mathbf{x}, \mathbf{x}') = k_{\text{unstruct}}(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')$ (see Appendix B.2).

Our expression for the structured kernel provides a concrete connection to the kernel theory of learning using nonlinear neural networks. For readers interested in such kernel theories, a full example and simulation results of how these work is given in Appendix B.3. There we show that there can be an exponential reduction in the number of samples needed to learn frequency detection using a structured versus unstructured basis.

4.3.3 Examples of random yet structured receptive fields

Our goal is to model the weights of artificial neurons in a way that is inspired by biological neurons' receptive fields. Structured RFNs sample hidden weights from GPs with structured covariance, so we construct covariance functions that make the generated weights resemble neuronal receptive fields. We start with a toy example of a stationary GP with well-understood Fourier eigenbasis and show how the receptive fields generated from this GP are selective to frequencies in timeseries signals. Then, we construct locally stationary covariance models of the of insect mechanosensory and V1 neuron receptive fields. These models are shown to be a good match for experimental data.

Warm-up: frequency selectivity from stationary covariance

To illustrate some results from our theoretical analysis, we start with a toy example of temporal receptive fields that are selective to particular frequencies. This example may be familiar to readers comfortable with Fourier series and basic signal processing. Let the input be a finite continuous timeseries $x(t)$ over the interval $T = [0, L]$. We use the covariance function

$$C(t, t') = \overbrace{\sum_{k=0}^{\infty} \lambda_k^2 \cos\left(\omega_k (t - t')\right)}^{\text{stationary process}}, \tag{4.5}$$

where $\omega_k = 2\pi k/L$ is the kth natural frequency and λ_k^2 are the weight coefficients. The covariance function Eq (4.5) is *stationary*, which means that it only depends on the difference between the timepoints $t - t'$. Applying the compound angle formula, we get

$$C(t,t') = \sum_{k=0}^{\infty} \lambda_k^2 \left(\cos(\omega_k t)\cos(\omega_k t') + \sin(\omega_k t)\sin(\omega_k t')\right). \tag{4.6}$$

Since the sinusoidal functions $\cos(\omega_k t)$ and $\sin(\omega_k t)$ form an orthonormal basis for $L^2(T)$, Eq (4.6) is the eigendecomposition of the covariance, where the eigenfunctions are sines and cosines with eigenvalues λ_k^2. From Eq (4.2), we know that structured weights with this covariance form a random series:

$$w(t) = \sum_{k=0}^{\infty} \lambda_k \left(z_k \cos(\omega_k t) + z_k' \sin(\omega_k t)\right), \tag{4.7}$$

where each $z_k, z_k' \sim \mathcal{N}(0,1)$. Thus, the receptive fields are made up of sinusoids weighted by λ_k and the Gaussian variables z_k, z_k'.

Suppose we want receptive fields that only retain specific frequency information of the signal and filter out the rest. Take $\lambda_k = 0$ for any k where $\omega_k < f_{\text{lo}}$ or $\omega_k > f_{\text{hi}}$. We call this a *bandlimited* spectrum with passband $[f_{\text{lo}}, f_{\text{hi}}]$ and bandwidth $f_{\text{hi}} - f_{\text{lo}}$. As the bandwidth increases, the receptive fields become less smooth since they are made up of a wider range of frequencies. If the λ_k are all nonzero but decay at a certain rate, this rate controls the smoothness of the resulting GP [222].

When these receptive fields act on input signals $x(t)$, they implicitly transform the inputs into the Fourier basis and filter frequencies based on the magnitude of λ_k. In a bandlimited setting, any frequencies outside the passband are filtered out, which makes the receptive fields selective to a particular range of frequencies and ignore others. On the other hand, classical random features weight all frequencies equally, even though in natural settings high frequency signals are the most corrupted by noise.

Insect mechanosensors

We next consider a particular biological sensor that is sensitive to the time-history of forces. Campaniform sensilla (CS) are dome-shaped mechanoreceptors that detect local stress and strain on the insect exoskeleton [62]. They are embedded in the cuticle and deformation of the cuticle through

bending or torsion induces depolarizing currents in the CS by opening mechanosensitive ion channels. The CS encode proprioceptive information useful for body state estimation and movement control during diverse tasks like walking, kicking, jumping, and flying [62].

We will model the receptive fields of CS that are believed to be critical for flight control, namely the ones found at the base of the halteres [233] and on the wings [173] (Fig 4.2A). Halteres and wings flap rhythmically during flight, and rotations of the insect's body induce torsional forces that can be felt on these active sensory structures. The CS detect these small strain forces, thereby encoding angular velocity of the insect body [233]. Experimental results show haltere and wing CS are selective to a broad range of oscillatory frequencies [76, 173], with STAs that are smooth, oscillatory, selective to frequency, and decay over time [77] (Fig 4.2B).

We model these temporal receptive fields with a locally stationary GP [84] with bandlimited spectrum. Examples of receptive fields generated from this GP are shown in Fig 4.2C. The inputs to the CS are modeled as a finite continuous timeseries $x(t)$ over the finite interval $T = [0, L]$. The neuron weights are generated from a covariance function

$$C(t,t') = \overbrace{\exp\left(-\frac{(t+t')}{\gamma}\right)}^{\text{localized}} \overbrace{\sum_{k=0}^{\infty} \lambda_k^2 \cos\left(\omega_k (t-t')\right)}^{\text{stationary process}}, \qquad \lambda_k = \overbrace{\begin{cases} 1 & f_{\text{lo}} \le \omega_k \le f_{\text{hi}} \\ 0 & \text{otherwise} \end{cases}}^{\text{bandlimited, flat-power spectrum}}, \tag{4.8}$$

where $\omega_k = 2\pi k/L$ is the kth natural frequency. As in the warmup, the frequency selectivity of their weights is accounted for by the parameters f_{lo} and f_{hi}. As the bandwidth $f_{\text{hi}} - f_{\text{lo}}$ increases, the receptive fields are built out of a wider selection of frequencies. This makes the receptive fields less smooth (Fig 4.2D). Each field is localized to near $t = 0$, and its decay with t is determined by the parameter γ. As γ increases, the receptive field is selective to larger time windows.

The eigenbasis of the covariance function Eq (4.8) is similar to a Fourier eigenbasis modulated by a decaying exponential. The eigenbasis is an orthonormal basis for the span of $\lambda_k e^{-t/\gamma}\cos(\omega_k t)$ and $\lambda_k e^{-t/\gamma}\sin(\omega_k t)$, which are a non-orthogonal set of functions in $L^2(T)$. The hidden weights transform timeseries inputs into this eigenbasis and discard components outside the passband frequencies $[f_{\text{lo}}, f_{\text{hi}}]$.

We fit the covariance model to receptive field data from 95 CS neurons from wings of the hawkmoth *Manduca sexta* (data from [173]). Briefly, CS receptive fields were estimated as the

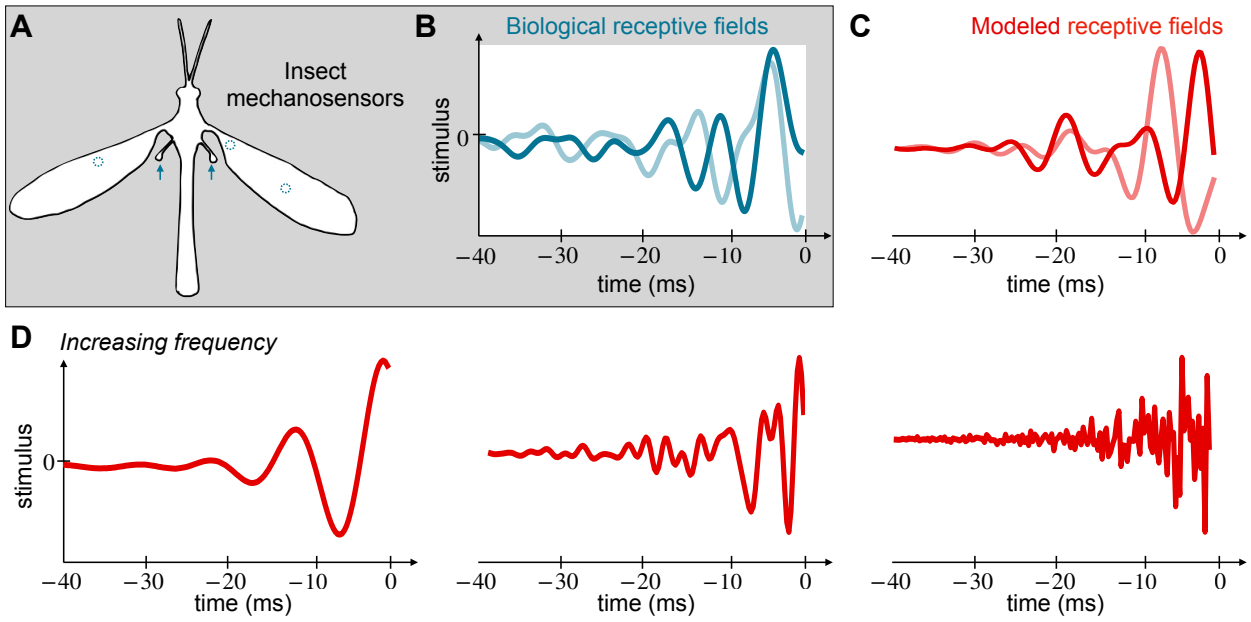

Figure 4.2: **Random receptive field model of insect mechanosensors.** (A) Diagram of the the cranefly, *Tipula hespera*. Locations of the mechanosensors, campaniform sensilla, are marked in blue on the wings and halteres. (B) Two receptive fields of campaniform sensilla are shown in blue. They are smooth, oscillatory, and decay over time. We model them as random samples from distributions parameterized by frequency and decay parameters. Data are from the hawkmoth [173]; cranefly sensilla have similar responses [77]. (C) Two random samples from the model distribution are shown in red. (D) The smoothness of the receptive fields is controlled by the frequency parameter. The decay parameter controls the rate of decay from the origin (not shown).

spike-triggered average (STA) of experimental mechanical stimuli of the wings, where the stimuli were generated as bandpassed white noise (2–300 Hz).

To characterize the receptive fields of this population of CS neurons, we compute the data covariance matrix $\mathbf{C}_{\text{data}}$ by taking the inner product between the receptive fields. We normalize the trace to be the dimension of each receptive field (number of samples), which in this case is 40

kHz $\times$ 40 ms = 1600 samples. This normalization sets the overall scale of the covariance matrix. The data covariance matrix shows a tridiagonal structure (Fig 4.3A). The main diagonal is positive while the off diagonals are negative. All diagonals decay away from the top left of the matrix.

To fit the covariance model to the data, we optimize the parameters $f_{\text{lo}}, f_{\text{hi}}$, and γ, finding $f_{\text{lo}} = 75$ Hz, $f_{\text{hi}} = 200$ Hz, and $\gamma = 12.17$ ms best fit the sensilla data. We do so by minimizing the Frobenius norm of the difference between $\mathbf{C}_{\text{data}}$ and the model (see Appendix B.4). The resulting model covariance matrix (Fig 4.3B) matches the data covariance matrix (Fig 4.3A) remarkably well qualitatively. The normalized Frobenius norm of the difference between $\mathbf{C}_{\text{data}}$ and the model is 0.4386. Examples of biological receptive fields and random samples from this fitted covariance model are shown in Figure B.2. To simulate the effect of a finite number of neurons, we generate 95 weight vectors (equal to the number of neurons recorded) and recompute the model covariance matrix (Fig 4.3C). We call this the finite neuron model covariance matrix $\mathbf{C}_{\text{finite}}$, and it shows the bump and blob-like structures evident in $\mathbf{C}_{\text{data}}$ but not in $\mathbf{C}_{\text{model}}$. This result suggests that these bumpy structures can be attributed to having a small number of recorded neurons. We hypothesize that these effects would disappear with a larger dataset and $\mathbf{C}_{\text{data}}$ would more closely resemble $\mathbf{C}_{\text{model}}$.

For comparison, we also calculate the Frobenius difference for null models, the unstructured covariance model and the Fourier model (B.20). For the unstructured model, the Frobenius norm difference is 0.9986 while that of the Fourier model is 0.9123. The sensilla covariance model has a much lower difference (0.4386) compared to the null models, fitting the data more accurately. We show the covariance matrices and sampled receptive fields from the null models in Figure B.3–Figure B.5.

Comparing the eigenvectors and eigenvalues of the data and model covariance matrices, we find that the spectral properties of both $\mathbf{C}_{\text{model}}$ and $\mathbf{C}_{\text{finite}}$ are similar to that of $\mathbf{C}_{\text{data}}$. The eigenvalue curves of the models match that of the data quite well (Fig 4.3E); these curves are directly comparable because each covariance is normalized by its trace, which makes the sum of the eigenvalues unity. Further, all of the data and the model covariance matrices are low-dimensional. The first 10 data eigenvectors explain 97% of the variance, and the top 5 explain 90%. The top 5 eigenvectors of the model and its finite sample match that of the data quite well (Fig 4.3D).

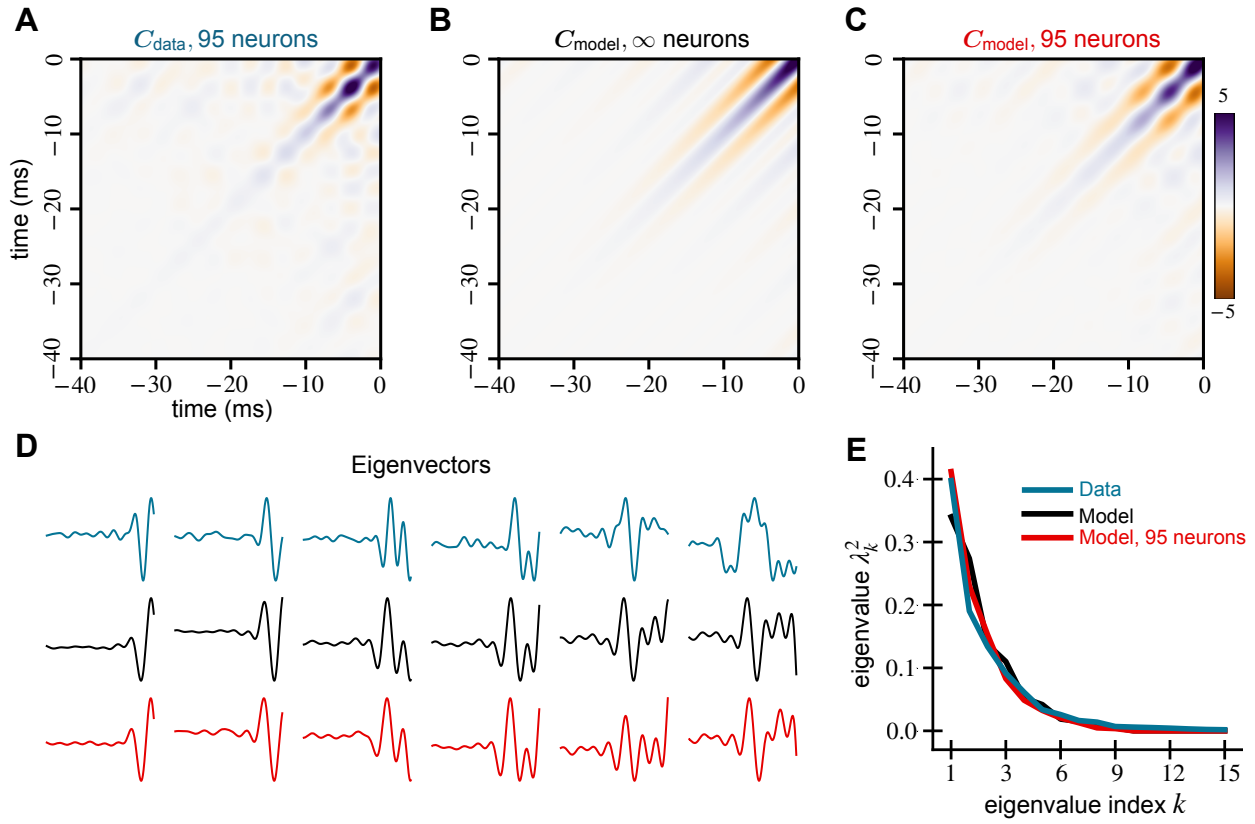

Figure 4.3: **Spectral properties of mechanosensory RFs and our model are similar.** We compare the covariance matrices generated from (A) receptive fields of 95 mechanosensors from [173], (B) the model Eq (4.8), and (C) 95 random samples from the same model. All covariance matrices show a tri-diagonal structure that decays away from the origin. (D) The first five principal components of all three covariance matrices are similar and explain 90% of the variance in the RF data. (E) The leading eigenvalue spectra of the data and models show similar behavior.

Primary visual cortex

We now turn to visually driven neurons from the mammalian primary cortex. Primary visual cortex (V1) is the earliest cortical area for processing visual information (Fig 4.4A). The neurons in V1

can detect small changes in visual features like orientations, spatial frequencies, contrast, and size.

Here, we model the receptive fields of simple cells in V1, which have clear excitatory and inhibitory regions such that light shone on the excitatory regions increase the cell's response and vice-versa (Fig 4.4B). The shape of the regions determines the orientation selectivity, while their widths determine the frequency selectivity. The receptive fields are centered to a location in the visual field and decay away from it. They integrate visual stimuli within a small region of this center [107]. Gabor functions are widely used as a mathematical model of the receptive fields of simple cells [116].

We model these receptive fields using another locally stationary GP [84] and show examples of generated receptive fields in Fig 4.4C. Consider the inputs to the cortical cells to be a continuous two-dimensional image $x(\mathbf{t})$, where the domain $T = [0, L] \times [0, L']$ and $x : T \to \mathbb{R}$. Since the image is real-valued, $x(\mathbf{t})$ is the grayscale contrast or single color channel pixel values. The neuron weights are then generated from a covariance function of the following form:

$$C(\mathbf{t}, \mathbf{t}') = \overbrace{\exp\left(-\frac{\|\mathbf{t}-\mathbf{t}'\|^2}{2f^2}\right)}^{\text{smooth receptive fields}} \cdot \overbrace{\exp\left(-\frac{\|\mathbf{t}-\mathbf{c}\|^2 + \|\mathbf{t}'-\mathbf{c}\|^2}{2s^2}\right)}^{\text{localized to a center } c}. \tag{4.9}$$

The receptive field center is defined by $\mathbf{c}$, and the size of the receptive field is determined by the parameter s. As s increases, the receptive field extends farther from the center $\mathbf{c}$ (Fig 4.4D). Spatial frequency selectivity is accounted for by the bandwidth parameter f. As f decreases, the spatial frequency of the receptive field goes up, making the weights less smooth (Fig 4.4E).

The eigendecomposition of the covariance function Eq (4.9) leads to an orthonormal basis of single scale *Hermite wavelets* [137, 138]. When $\mathbf{c} = 0$, the wavelet eigenfunctions are Hermite polynomials modulated by a decaying Gaussian:

$$\phi_{\mathbf{k}}(\mathbf{t}) \propto \prod_{i=1}^{D} e^{-c_1 {t_i}^2} H_{k_i}(c_2 t_i) \quad \text{and} \quad \lambda_{\mathbf{k}}^2 \propto \prod_{i=1}^{D} c_3^{k_i}, \tag{4.10}$$

where H_k is the kth (physicist's) Hermite polynomial; eigenfunctions for nonzero centers $\mathbf{c}$ are just shifted versions of Eq (4.10). The detailed derivation and values of the constants c_1, c_2, c_3 and normalization are in Appendix B.6.

We use Eq (4.9) to model receptive field data from 8,358 V1 neurons recorded with calcium imaging from transgenic mice expressing GCaMP6s; the mice were headfixed and running on an air-

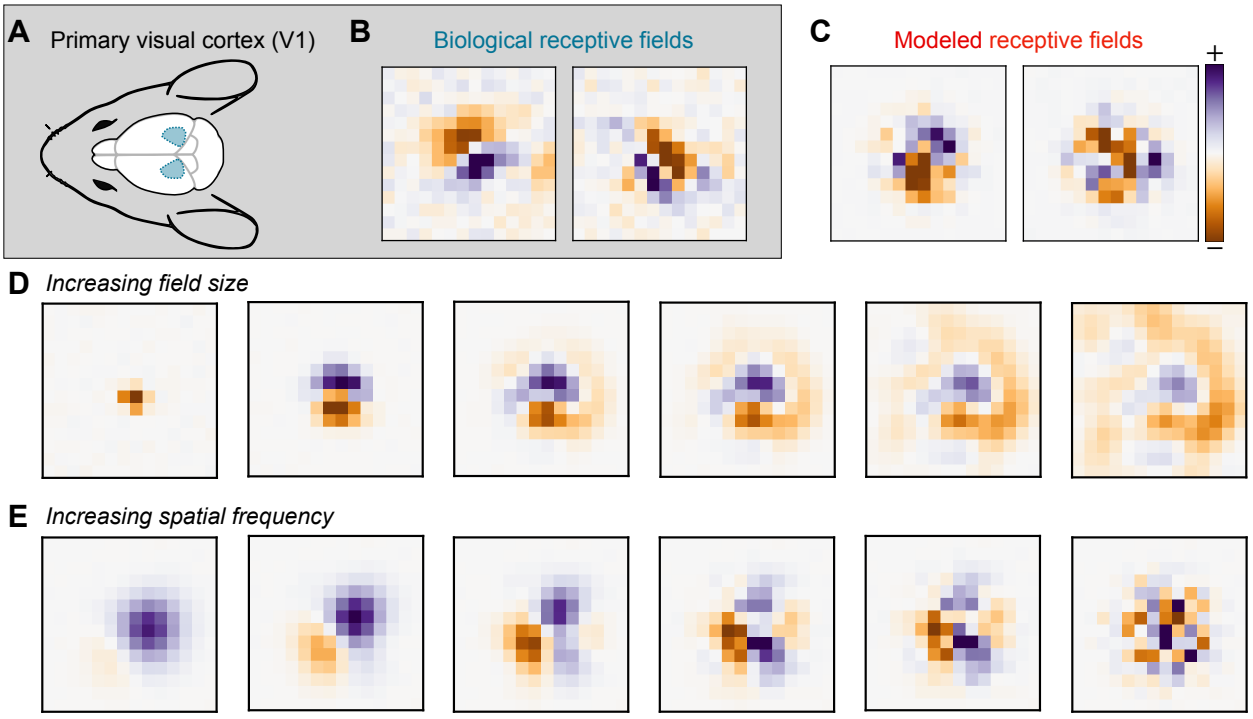

Figure 4.4: **Random receptive field model of Primary Visual Cortex (V1).** (A) Diagram of the mouse brain with V1 shown in blue. (B) Receptive fields of two mouse V1 neurons calculated from their response to white noise stimuli. The fields are localized to a region in a visual field and show "on" and "off" regions. (C) Random samples from the model Eq (4.9) distribution. (D) Increasing the receptive field size parameter in our model leads to larger fields. (E) Increasing the model spatial frequency parameter leads to more variable fields.

floating ball. We presented 24,357 unique white noise images of 14×36 pixels using the Psychtoolbox [120], where the pixels were white or black with equal probability. Images were upsampled to the resolution of the screens via bilinear interpolation. The stimulus was corrected for eye-movements online using custom code. The responses of 45,026 cells were collected using a two-photon mesoscope [202] and preprocessed using Suite2p [158]. Receptive fields were calculated from the white

noise images and the deconvolved calcium responses of the cells using the STA. For the covariance analysis, we picked cells above the signal-to-noise (SNR) threshold of 0.4; this gave us 8,358 cells. The SNR was calculated from a smaller set of 2,435 images that were presented twice using the method from [208]. As a preprocessing step, we moved the center of mass of every receptive field to the center of the visual field.

We compute the data covariance matrix $\mathbf{C}_{\text{data}}$ by taking the inner product between the receptive fields. We normalize the trace to be the dimension of each receptive field, which in this case is (14 $\times$ 36) pixels = 504 pixels. The data covariance matrix resembles a tridiagonal matrix. However, the diagonals are non-zero only at equally spaced segments. Additionally, their values decay away from the center of the matrix. We show $\mathbf{C}_{\text{data}}$ zoomed in at the non-zero region around the center of the matrix (Fig 4.5A); this corresponds to the 180 $\times$ 180 pixel region around the center of the full 504 $\times$ 504 pixel matrix. The full covariance matrix is shown in Figure B.6.

In the covariance model, the number of off-diagonals, the center, the rate of their decay away from the center are determined by the parameters f, s and $\mathbf{c}$ respectively. The covariance between pixels decays as a function of their distance from $\mathbf{c}$. This leads to the the equally-spaced non-zero segments. On the other hand, the covariance also decays as a function of the distance between pixels. This brings the the diagonal structure to the model. When the frequency parameter f increases, the number of off-diagonals increases. Pixels in the generated weights become more correlated and the weights become spatially smoother. When the size parameter s increases, the diagonals decay slower from the center $\mathbf{c}$, increasing correlations with the center pixel and leading the significant weights to occupy more of the visual field.

We again optimize the parameters to fit the data, finding $s = 1.87$ and $f = 0.70$ pixels, by minimizing the Frobenius norm of the difference between $\mathbf{C}_{\text{data}}$ and the model. We do not need to optimize over the center parameter $\mathbf{c}$, since we preprocess the data so that all receptive fields are centered at $\mathbf{c} = (7, 18)$, the center of the 14$\times$36 grid. The resulting model covariance matrix (Fig 4.5B) and the data covariance matrix (Fig 4.5A) match remarkably well qualitatively. The normalized Frobenius norm of the difference between $\mathbf{C}_{\text{data}}$ and the model is 0.2993. Examples of biological receptive fields and random samples from the fitted covariance model are shown in Figure B.7. To simulate the effect of a finite number of neurons, we generate 8,358 weights, equal to

the number of neurons in our data, to compute $\mathbf{C}_{\text{finite}}$ shown in Fig 4.5C. This finite matrix $\mathbf{C}_{\text{finite}}$ looks even more like $\mathbf{C}_{\text{data}}$, and it shows that some of the negative covariances far from center result from finite sample size but not all.

For comparison, we also calculate the normalized Frobenius difference for null models, the unstructured covariance model and a translation invariant V1 model. In the translation invariant model, we remove the spatially localizing exponential in Eq (4.9) and only fit the spatial frequency parameter, f. For the unstructured model, the Frobenius norm difference is 0.9835 while that of the translation invariant model is 0.9727. The V1 covariance model has a much lower difference (0.2993) and is a better fit to the data. We show the covariance matrices and sampled receptive fields from these null models in Figure B.8–Figure B.10.

Similar spectral properties are evident in the eigenvectors and eigenvalues of $\mathbf{C}_{\text{model}}$, $\mathbf{C}_{\text{finite}}$, $\mathbf{C}_{\text{data}}$, and the analytical forms derived in Eq (4.10) (Fig 4.5D-E). The covariances are again normalized to have unit trace. Note that the analytical eigenfunctions are shown on a finer grid than the model and data because the analysis was performed in continuous space. The differences between the eigenfunctions and eigenvalues of the analytical and model results are due to discretization. Examining the eigenvectors (Fig 4.5D), we also see a good match, although there are some rotations and differences in ordering. These 10 eigenvectors explain 68% of the variance in the receptive field data. For reference, the top 80 eigenvectors explain 86% of the variance in the data and all of the variance in the model. The eigenvalue curves of both the models and the analytical forms match that of the data (Fig 4.5E) reasonably well, although not as well as for the mechanosensors. In Appendix B.11, we repeat this analysis for receptive fields measured with different stimulus sets in the mouse and different experimental dataset from non-human primate V1. Our findings are consistent with the results shown above (Figure B.11–Figure B.16).

4.3.4 Advantages of structured random weights for artificial learning tasks

Our hypothesis is that neuronal inductive bias from structured receptive fields allows networks to learn with fewer neurons, training examples, and steps of gradient descent for classification tasks with naturalistic inputs. To examine this hypothesis, we compare the performance of structured receptive fields against classical ones on several classification tasks. We find that, for most artificial

learning tasks, structured random networks learn more accurately from smaller network sizes, fewer training examples, and gradient steps.

Frequency detection

CS naturally encode the time-history of strain forces acting on the insect body and sensors inspired by their temporal filtering properties have been shown to accurately classify spatiotemporal data [145]. Inspired by this result, we test sensilla-inspired mechanosensory receptive fields on a timeseries classification task (Fig 4.6A, top). Each example presented to the network is a 100 ms timeseries sampled at 2 kHz so that $d = 200$, and the goal is to detect whether or not each example contains a sinusoidal signal. The positive examples are sinusoidal signals with $f_1 = 50$ Hz and corrupted by noise so that their SNR = 1.76 (2.46 dB). The negative examples are Gaussian white noise with matched amplitude to the positive examples. Note that this frequency detection task is not linearly separable because of the random phases in positive and negative examples. See Appendix B.8 for additional details including the definition of SNR and how cross-validation was used to find the optimal parameters $f_{\text{lo}} = 10$ Hz, $f_{\text{hi}} = 60$ Hz, and $\gamma = 50$ ms.

For the same number of hidden neurons, the structured RFN significantly outperforms a classical RFN. We show test performance using these tuned parameters in Fig 4.6A. Even in this noisy task, it achieves 0.5% test error using only 25 hidden neurons. Meanwhile, the classical network takes 300 neurons to achieve similar error.

Predictably, the performance suffers when the weights are *incompatible* with the task. We show results when $f_{\text{lo}} = 10$ Hz and $f_{\text{hi}} = 40$ Hz and the same γ (Fig 4.6A). The incompatible RFN performs better than chance (50% error) but much worse than the classical RFN. It takes 300 neurons just to achieve 16.3% test error. The test error does not decrease below this level even with additional hidden neurons.

Frequency XOR task

To challenge the mechanosensor-inspired networks on a more difficult task, we build a frequency Exclusive-OR (XOR) problem (Fig 4.6B, top). XOR is a binary function which returns true if and only if the both inputs are different, otherwise it returns false. XOR is a classical example of

a function that is not linearly separable and thus harder to learn. Our inputs are again 100 ms timeseries sampled at 2 kHz. The inputs either contain a pure frequency of $f_1 = 50$ Hz or $f_2 = 80$ Hz, mixed frequency signals with both f_1 and f_2, or white noise. In both the pure and mixed frequency cases, we add noise so that the SNR = 1.76. See Appendix B.8 for details. The goal of the task is to output true if the input contains either pure tone and false if the input contains mixed frequencies or is white noise.

We tune the GP covariance parameters f_{lo}, f_{hi}, and γ from Eq (4.8) using cross-validation. The cross validation procedure and algorithmic details are identical to that of the frequency detection task. Using cross validation, we find the optimal parameters to be $f_{\text{lo}} = 50$ Hz, $f_{\text{hi}} = 90$ Hz, and $\gamma = 40$ ms. For incompatible weights, we take $f_{\text{lo}} = 10$ Hz, $f_{\text{hi}} = 60$ Hz, and the same γ.

The structured RFN significantly outperform classical RFN for the same number of hidden neurons. We show network performance using these parameters in Fig 4.6B. Classification error of 1% can be achieved with 25 hidden neurons. In sharp contrast, the classical RFN requires 300 hidden neurons just to achieve 6% error. With incompatible weights, the network needs 300 neurons to achieve just 15.1% test error and does not improve with larger network sizes. Out of the four input subclasses, it consistently fails to classify pure 80 Hz sinusoidal signals which are outside its passband.

Image classification

We next test the V1-inspired receptive fields on two standard digit classification tasks, MNIST [128] and KMNIST [51]. The MNIST and KMNIST datasets each contain 70,000 images of handwritten digits. In MNIST, these are the Arabic numerals 0–9, whereas KMNIST has 10 Japanese *hiragana* phonetic characters. Both datasets come split into 60,000 training and 10,000 test examples. With 10 classes, there are 6,000 training examples per class. Every example is a 28×28 grayscale image with centered characters.

Each hidden weight has its center $\mathbf{c}$ chosen uniformly at random from all pixels. This ensures that the network's weights uniformly cover the image space and in fact means that the network can represent any sum of locally-smooth functions (see Appendix B.7). We use a network with 1,000 hidden neurons and tune the GP covariance parameters s and f from Eq (4.9) using 3-fold cross

validation on the MNIST training set. Each parameter ranges from 1 to 20 pixels, and the optimal parameters are found with a grid search. We find the optimal parameters to be $s = 5$ pixels and $f = 2$ pixels. We then refit the optimal model using the entire training set. The parameters from MNIST were used on the KMNIST task without additional tuning.

The V1-inspired achieves much lower average classification error as compared to the classical RFN for the same number of hidden neurons. We show learning performance using these parameters on the MNIST task in Fig 4.7A. To achieve 6% error on the MNIST task requires 100 neurons versus 1,000 neurons for the classical RFN, and the structured RFN achieves 2.5% error with 1,000 neurons. Qualitatively similar results hold for the KMNIST task (Fig 4.7B), although the overall errors are larger, reflecting the harder task. To achieve 28% error on KMNIST requires 100 neurons versus 1,000 neurons for the classical RFN, and the structured RFN achieves 13% error with 1,000 neurons.

Again, network performance suffers when GP covariance parameters do not match the task. This happens if the size parameter s is smaller than the stroke width or spatial scale f doesn't match the stroke variations in the character. Taking the incompatible parameters $s = 0.5$ and $f = 0.5$ (Fig 4.7A-B), the structured weights performs worse than the classical RFN in both tasks. With 1,000 hidden neurons, it achieves the relatively poor test errors of 8% on MNIST (Fig 4.7A) and 33% on KMNIST (Fig 4.7B).

Structured weights improve generalization with limited data

Alongside learning with fewer hidden neurons, V1 structured RFNs also learn more accurately from fewer examples. We test few-shot learning using the image classification datasets. The training examples are reduced from 60,000 to 50, or only 5 training examples per class. The test set and GP parameters remain the same.

Structured encodings allow learning with fewer samples than unstructured encodings. We show these few-shot learning results in Fig 4.7C-D. The networks' performance saturate past a few hundred hidden neurons. For MNIST, the lowest error achieved by V1 structured RFN is 27% versus 33% for the classical RFN and 37% using incompatible weights (Fig 4.7C). The structured network acheives 61% error using structured features on the KMNIST task, as opposed to 66% for

the classical RFN and 67% using incompatible weights (Fig 4.7D).

Networks train faster when initialized with structured weights

Now we study the effect of structured weights as an initialization strategy for fully-trained neural networks where all weights in the network vary. We hypothesized that structured initialization allows networks to learn faster, i.e. that the training loss and test error would decrease faster than with unstructured weights. We have shown that the performance of RFNs improves with biologically inspired weight sampling. However, in RFNs Eq (4.1) only the readout weights β are modified with training, and the hidden weights $\mathbf{W}$ are frozen at their initial value.

We compare the biologically-motivated initialization with a classical initialization where the variance is inversely proportional to the number of hidden neurons, $\mathbf{w}_{\text{unstruct}} \sim \mathcal{N}(0, \frac{2}{d}\mathbf{I})$. This initialization is widely known as the "Kaiming He normal" scheme and is thought to stabilize training dynamics by controlling the magnitude of the gradients [103]. The classical approach ensures that $\text{Tr}(\frac{2}{d}\mathbf{I}) = 2$, so for fair comparison we scale our structured weight covariance matrix to have $\text{Tr}(\mathbf{C}) = 2$. In our studies with RFNs the trace is equal to d, but this weight scale can be absorbed into the readout weights β due to the homogeneity of the ReLU.

We again compare structured and unstructured weights on MNIST and KMNIST tasks, common benchmarks for fully-trained networks. The architecture is a single hidden layer feedforward neural network (Fig 4.1) with 1,000 hidden neurons. The cross-entropy loss over the training sets are minimized using simple gradient descent (GD) for 3,000 epochs. For a fair comparison, the learning rate is optimized for each network separately. We define the area under the training loss curve as a metric for the speed of learning. Then, we perform a grid search in the range of $(1e^{-4}, 1e^{0})$ for the learning rate that minimizes this metric, resulting in the parameters $0.23, 0.14, 0.14$ for structured, unstructured and incompatible networks respectively. All other parameters are the same as for image classification.

In both the MNIST and KMNIST tasks, the V1-initialized network minimizes the loss function faster than the classically initialized network. For the MNIST task, the V1 network achieves a loss value of 0.05 after 3,000 epochs compared to 0.09 for the other network (Fig 4.8A). We see qualitatively similar results for the KMNIST task. At the end of training, the V1-inspired

network's loss is 0.08, while the classically initialized network only reaches 0.12 (Fig 4.8B). We find that the the V1-initialized network performs no better than classical initialization when the covariance parameters do not match the task. With incompatible parameters, the V1-initialized network achieves a loss value of 0.11 on MNIST and 0.15 on KMNIST.

Not only does it minimize the training loss faster, the V1-initialized network also generalizes well and achieves a lower test error at the end of training. For MNIST, it achieves 1.7% test error compared to 3.3% error for the classically initialized network, and 3.6% using incompatible weights (Fig 4.8C). For KMNIST, we see 9% error compared to 13% error with classical initalization and 15% using incompatible weights (Fig 4.8D).

We see similar results across diverse hidden layer widths and learning rates (Figure B.17–Figure B.20), with the benefits most evident for wider networks and smaller learning rates. Furthermore, the structured weights show similar results when trained for 10,000 epochs (rate 0.1; 1,000 neurons; not shown) and with other optimizers like minibatch Stochastic Gradient Descent (SGD) and ADAM (batch size 256, rate 0.1; 1,000 neurons; not shown). Structured initialization facilitates learning across a wide range of networks.

However, the improvement is not universal: no significant benefit was found by initializing the early convolutional layers of the deep network AlexNet [126] and applying it to the ImageNet dataset [189], as shown in Appendix B.13 and Figure B.21. The large amounts of training data and the fact that only a small fraction of the network was initialized with structured weights could explain this null result. Also, in many of these scenarios the incompatible structured weights get to performance on par with the compatible ones by the end of training, when the poor inductive bias is overcome.

Improving representation with structured random weights

We have shown how structured receptive field weights can improve the performance of RFNs and fully-trained networks on a number of supervised learning tasks. As long as the receptive fields are compatible with the task itself, then performance gains over unstructured features are possible. If they are incompatible, then the networks performs no better or even worse than using classical unstructured weights.

These results can be understood with our theoretical framework. Structured weights effectively cause the input $\mathbf{x}$ to undergo a linear transformation into a new representation $\tilde{\mathbf{x}}$ following Theorem 4.1. In all of our examples, this new representation is bandlimited due to how we design the covariance function. (The V1 weights have all eigenvalues nonzero, but the spectrum decays exponentially, so it acts as a lowpass filter.) By moving to a bandlimited representation, we both filter out noise—high-frequency components—and reduce dimensionality—coordinates in $\tilde{\mathbf{x}}$ outside the passband are zero. In general, noise and dimensionality both make learning harder.

It is easiest to understand these effects in the frequency detection task. For simplicity, assume we are using the stationary features of our warm-up to do frequency detection. In this task, all of the signal power is contained in the $f_1 = 50$ Hz frequency, and everything else is due to noise. If the weights are compatible with the task, this means that $\mathbf{w}$ is a sum of sines and cosines of frequencies ω_k in some passband which includes f_1. The narrower we make this bandwidth while still retaining the signal, the higher the SNR of $\tilde{\mathbf{x}}$ becomes since more noise is filtered out (see Appendix B.9).

4.4 Concluding remarks

In this article, we describe a random generative model for the receptive fields of sensory neurons. Specifically, we model each receptive field as a random filter sampled from a Gaussian process (GP) with covariance structure matched to the statistics of experimental neural data. We show that two kinds of sensory neurons—insect mechanosensory and simple cells in mammalian V1—have receptive fields that are well-described by GPs. In particular, the generated receptive fields, their second-order statistics, and their principal components match with receptive field data. Theoretically, we show that individual neurons perform a randomized transformation and filtering on the inputs. This connection provides a framework for sensory neurons to compute input transformations like Fourier and wavelet transforms in a biologically plausible way.

Our numerical results using these structured random receptive fields show that they offer better learning performance than unstructured receptive fields on several benchmarks. The structured networks achieve higher test performance with fewer neurons and fewer training examples, unless the frequency content of their receptive fields is incompatible with the task. In networks that are fully trained, initializing with structured weights leads to better network performance (as measured

by training loss and generalization) in fewer iterations of gradient descent. Structured random features may be understood theoretically as transforming inputs into an informative basis that retains the important information in the stimulus while filtering away irrelevant signals.

Modeling other sensory neurons and modalities

The random feature formulation is a natural extension of the traditional linear-nonlinear (LN) neuron model. This approach may be applied to other brain regions where LN models are successful, for instance sensory areas with primarily feedforward connectivity like somatosensory and auditory regions. The neurons in auditory and somatosensory systems are selective to both spatial and temporal structures in their stimuli [121, 191, 174], and spatial structure emerges in networks trained on artificial tactile tasks [243]. Their receptive fields could be modeled by GPs with spatiotemporal covariance functions [231]; these could be useful for artificial tasks with spatiotemporal stimuli such as movies and multivariate timeseries. Neurons with localized but random temporal responses were found to be compatible with manifold coding in a decision-making task [122]. Our GPs are a complementary approach to traditional sparse coding [153] and efficient coding [15, 45] hypotheses; the connections to these other theories are interesting for future research.

4.4.1 Receptive fields in development

Our generative model offers new directions to explore the biological basis and computational principles behind receptive fields. Development lays a basic architecture that is conserved from animal to animal [211, 207], yet the details of every neural connection cannot be specified [238], leading to some amount of inevitable randomness at least initially [44]. If receptive fields are random with constrained covariance, it is natural to ask how biology implements this. Unsupervised Hebbian dynamics with local inhibition can allow networks to learn principal components of their input [151, 169]. An interesting future direction is how similar learning rules may give rise to overcomplete, nonorthogonal structure similar to what has been studied here. This may prove more biologically plausible than weights that result from task-driven optimization.

The above assumes that receptive field properties actually lie within synaptic weights. For spatial receptive fields, this assumption is plausible [184], but the temporal properties of receptive

fields are more likely a result of neurons' intrinsic dynamics, for which the LN framework is just a model [155, 226, 67]. Heterogeneous physiological (e.g. resonator dynamics) and mechanical (position and shape of mechanosensor relative to body structure) properties combine to give the diverse temporal receptive field structures [17]. Development thus leverages different mechanisms to build structure into receptive field properties of sensory neurons.

4.4.2 Connections to compressive sensing

Random projections have seen extensive use in the field of compressive sensing, where a high-dimensional signal can be found from only a few measurements so long as it has a sparse representation [66, 73, 81]. Random compression matrices are known to have optimal properties, however in many cases structured randomness is more realistic. Recent work has shown that structured random projections with local wiring constraints (in one dimension) were compatible with dictionary learning [68], supporting previous empirical results [16]. Our work shows that structured random receptive fields are equivalent to employing a wavelet dictionary and dense Gaussian projection.

4.4.3 Machine learning and inductive bias

An important open question for both neuroscience and machine learning is why certain networks, characterized by features such as their architecture, weights, and nonlinearities, are better than others for certain problems. One perspective is that a network is good for a problem if it is biased towards approximating functions that are close to the target, known as an *inductive bias*, which depends on an alignment between the features encoded by neurons and the task at hand [36]. Our approach shows that structured receptive fields are equivalent to a linear transformation of the input that can build in such biases. Furthermore, we can describe the nonlinear properties of the network using the kernel, which varies depending on the receptive field structure. If the target function has a small norm in this RKHS, then there is an inductive bias and it is easier to learn [197, 196]. A small norm in the RKHS means that the target function varies smoothly over the inputs. Smooth functions are easier to learn compared to fast varying ones. In this way, the receptive field structure influences the ease of learning of the target function. We conjecture that receptive fields from neural-inspired distributions affect the RKHS geometry such that the target

function's norm is small in that RKHS, compared to the RKHS of random white-noise receptive fields. We leave to future work to verify this conjecture in detail.

Networks endowed with principles of neural computation like batch normalization, pooling of inputs, and residual connections have been found to contain inductive biases for certain learning problems [232, 102]. Learning data-dependent kernels is another way to add in inductive bias [201]. We also saw that initializing fully-trained networks from our generative models improved their speed of convergence and generalization compared to unstructured initialization. This result is consistent with known results that initialization has an effect on generalization [10]. The initialization literature has mostly been focused on avoiding exploding/vanishing gradients [103, 86]. Here, we conjecture that the inductive bias in our structured connectivity places the network closer to a good solution in the loss landscape [238].

The random V1-inspired receptive fields that we model can be seen as similar to what happens in a convolutional neural network (CNN) [152], which have similarities and differences compared to brains [132]. A recent study showed that CNNs with a fixed V1-like convolutional layer are more robust to adversarial perturbations to their inputs [57]. In a similar vein to our work, using randomly sampled Gabor receptive fields in the first layer of a deep network was also shown to improve its performance [110]. The wavelet scattering transform is a multi-layer network where wavelet coefficients are passed through nonlinearities, a model which is similar to deep CNNs [135, 39, 5]. Our framework differs as a randomized model and yields wavelets of a single scale, and similar studies of robustness and learning in deep networks with our weights are possible. Adding layers to our model or sampling weights with a variety of spatial frequencies and field sizes would yield random networks that behave similar to the scattering transform, offering an another connection between the brain and CNNs. Directly learning filters in a Hermite wavelet basis led to good perfomance in ANNs with little data [113], and this idea was extended to multiple scales by [171]. Our structured random features can be seen as an RFN version of those ideas with supporting evidence that these principles are used in biology.

4.4.4 Limitations and future directions

There are several limitations to the random feature approach. We model neuron responses with a scalar firing rates instead of discrete spikes, and we ignore complex neuronal dynamics, neuromodulatory context, and many other details. Like most LN models, the random feature model assumes zero plasticity in the hidden layer neurons. However, associative learning can drive changes in receptive fields of individual neurons in sensory areas like V1 and auditory cortex [88, 78]. Further, our RFN is purely feedforward and cannot account for feedback connections. Recent work suggests that feedforward architecture lacks sufficient computational power to serve as a detailed input-output model for a network of cortical neurons; it might need additional layers with convolutional filters [23]. It can be difficult to interpret the parameters found from fitting receptive field data and connect them to experimental conditions. Also, the GP model of weights only captures covariance (second moments) and neglects higher-order statistics. It remains to be shown how the theory can yield concrete predictions that can be tested *in vivo* experimental conditions.

The random feature receptive field model is a randomized extension of the LN neuron model. The LN model fits a parameterized function to each receptive field [49]. In contrast, the random feature framework fits a distribution to an entire population of receptive fields and generates realistic receptive fields from that distribution. A natural question is how they compare. If the goal is to capture individual differences between neuronal receptive fields, one should resort to an LN model where each neuron's receptive field is fit to data. The random feature model is not as flexible, but it provides a direct connection to random feature theory, and it is mathematically tractable and generative. This connection to kernel learning opens the door to using techniques which are a mainstay in machine learning theory literature, for instance to estimate generalization error and sample complexity [197], in the context of learning in more biologically realistic networks.

We see several future directions of structured random features in connecting computational neuroscience and machine learning. As already stated, the auditory, somatosensory, and tactile regions are good candidates for further study as well as developmental principles that could give rise to random yet structured receptive field properties. To account for plasticity in the hidden layer, one could also analyze the neural tangent kernel (NTK) associated with structured features [114]. These kernels are often used to analyze ANNs trained with gradient descent when the number of hidden

neurons is large and the step size is small [8]. To incorporate lateral and feedback connections, the weights could be sampled from GPs with recurrent covariance functions [143]. Our theory may also help explain why CNNs with fixed V1-like convolutional layer are more robust to adversarial input perturbations [57] as filtering out high frequency corruptions. It seems likely that structured random features will also be more robust. It would be interesting to study intermediate layer weights of fully-trained networks as approximate samples from a GP by studying their covariance structure. Finally, one could try and develop other covariance functions and further optimize these RFNs for most sophisticated learning tasks to see if near high performance—lower error, faster training, etc.—on more difficult tasks is possible.

4.5 Methods

The methods are described throughout the Results section. Further details and additional results are in Appendix B.

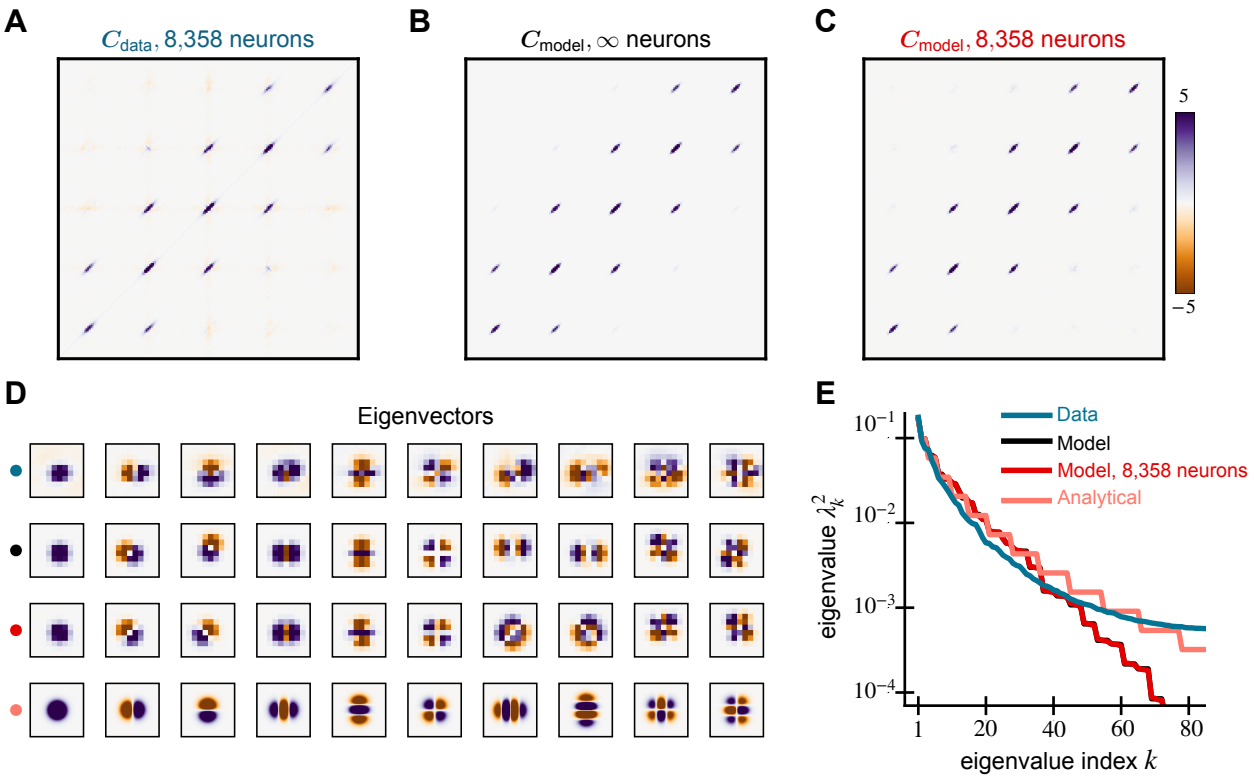

Figure 4.5: **Spectral properties of V1 RFs and our model are similar.** We compare the covariance matrices generated from the (A) receptive fields of 8,358 mouse V1 neurons, (B) the GP model Eq (4.9), and (C) 8,358 random samples from the model. These resemble a tri-diagonal matrix whose diagonals are non-zero at equally-spaced segments. (D) The leading 10 eigenvectors of the data and model covariance matrices show similar structure and explain 68% of the variance in the data. Analytical Hermite wavelet eigenfunctions are in the last row and differ from the model due to discretization (both cases) and finite sampling (8,358 neurons only). (E) The eigenspectrum of the model matches well with the data. The staircase pattern in the model comes from repeated eigenvalues at each frequency. The model curve with infinite neurons (black) is obscured by the model curve with 8,358 neurons (red).

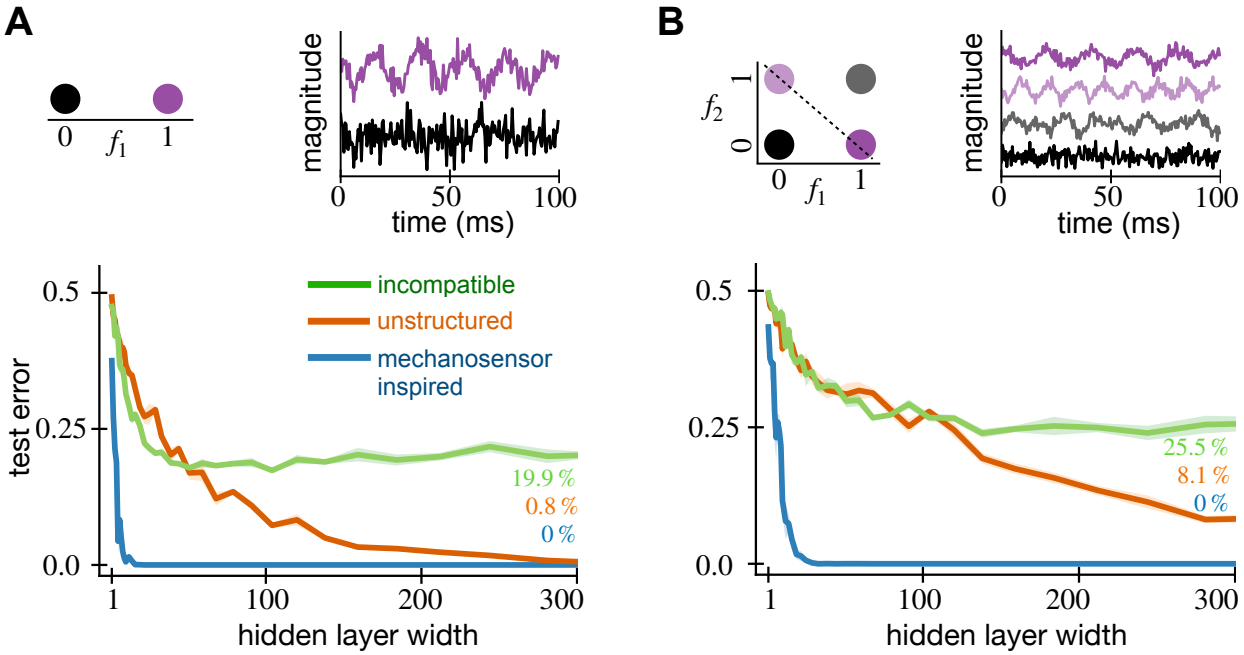

Figure 4.6: **Random mechanosensory weights enable learning with fewer neurons in time-series classification tasks.** We show the test error of random feature networks with both mechanosensory and classical white-noise weights against the number of neurons in their hidden layer. For every hidden layer width, we generate five random networks and average their test error. In the error curves, the solid lines show the average test error while the shaded regions represent the standard error across five generations of the random network. The top row shows the timeseries tasks that the networks are tested on. (A, top) In the frequency detection task, a $f_1 = 50$ Hz frequency signal (purple) is separated from white noise (black). (B, top) In the frequency XOR task, $f_1 = 50$ Hz (purple) and $f_2 = 80$ Hz (light purple) signals are separated from white noise (black) and mixtures of 50 Hz and 80 Hz (gray). When their covariance parameters are tuned properly, mechanosensor-inspired networks achieve lower error using fewer hidden neurons on both frequency detection (A, bottom) and frequency XOR (B, bottom) tasks. However, the performance of bio-inspired networks suffer if their weights are incompatible with the task.

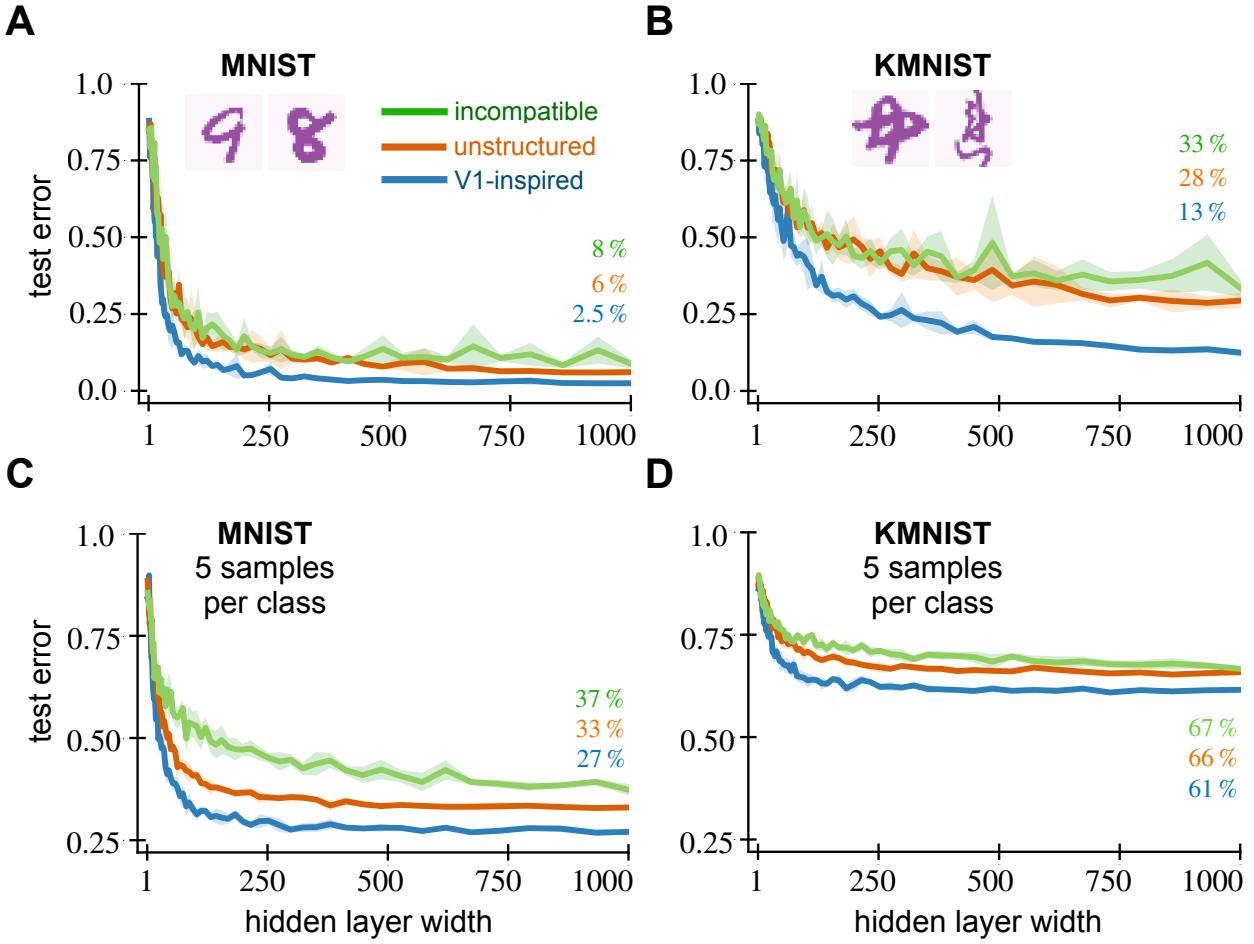

Figure 4.7: **Random V1 weights enable learning with fewer neurons and fewer examples on digit classification tasks.** We show the average test error of random feature networks with both V1 and classical white-noise weights against the number of neurons in their hidden layer. For every hidden layer width, we generate five random networks and average their test error. The solid lines show the average test error while the shaded regions represent the standard error across five generations of the random network. The top row shows the network's test error on (A) MNIST and (B) KMNIST tasks. When their covariance parameters are tuned properly, V1-inspired networks achieve lower error using fewer hidden neurons on both tasks. The network performance deteriorates when the weights are incompatible to the task. (C) MNIST and (D) KMNIST with 5 samples per class. The V1 network still achieves lower error on these fewshot tasks when the parameters are tuned properly.

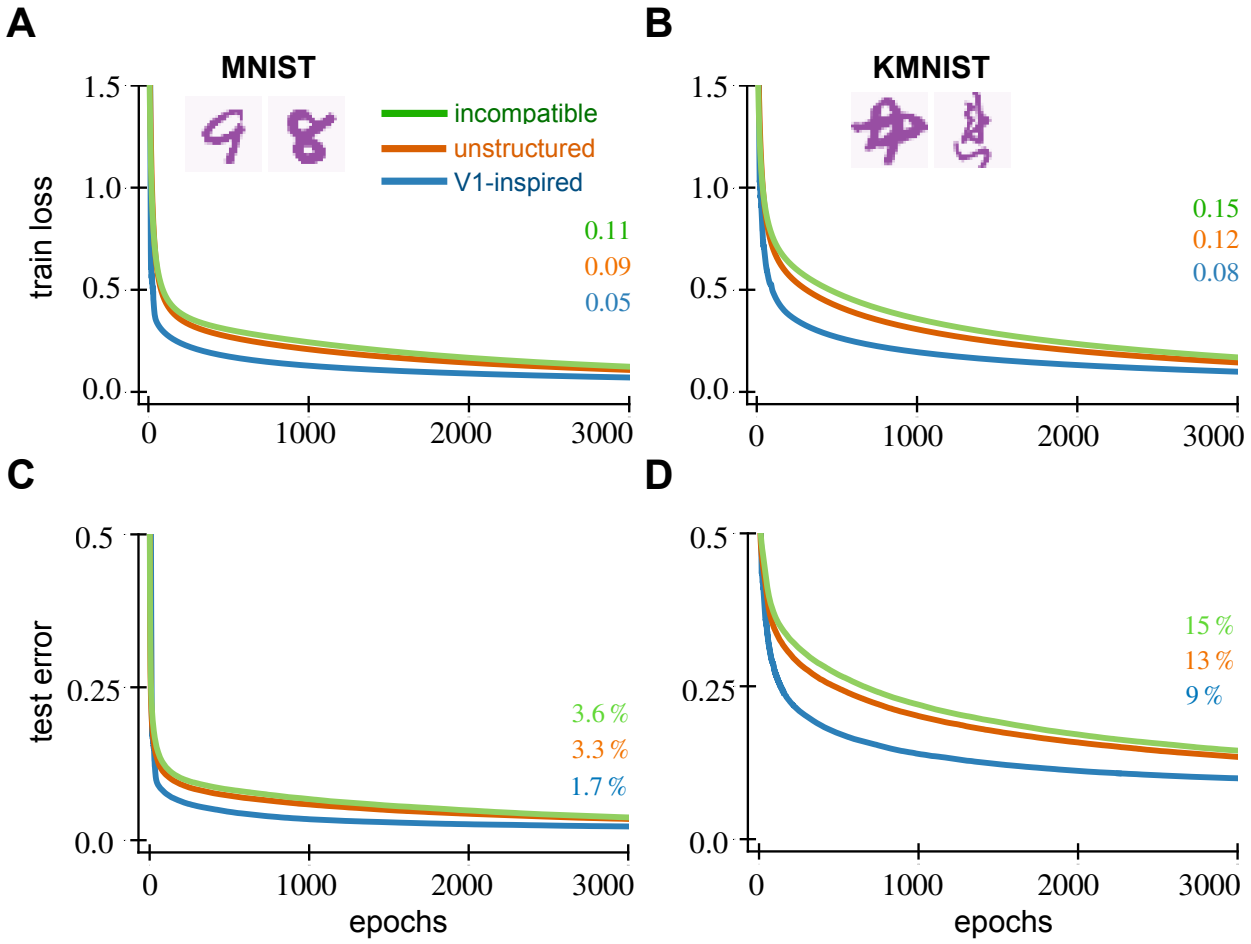

Figure 4.8: **V1 weight initialization for fully-trained networks enables faster training on digit classification tasks.** We show the average test error and the train loss of fully-trained neural networks against the number of training epochs. The hidden layer of each network contains 1,000 neurons. We generate five random networks and average their performance. The solid lines show the average performance metric across five random networks while the shaded regions represent the standard error. The top row shows the network's training loss on (A) MNIST and (B) KMNIST tasks. The bottom row shows the corresponding test error on (C) MNIST and (D) KMNIST tasks. When their covariance parameters are tuned properly, V1-initialized networks achieve lower training loss and test error under fewer epochs on both MNIST and KMNIST tasks. The network performance is no better than unstructured initialization when the weights are incompatible with the task.

BIBLIOGRAPHY

[1] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. U.S. Government Printing Office, 1964.

[2] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv:2303.08797*, 2023.

[3] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv:2209.15571*, 2022.

[4] Mauricio A Alvarez, Lorenzo Rosasco, Neil D Lawrence, et al. Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4(3):195–266, 2012.

[5] Joakim Andén and Stéphane Mallat. Deep Scattering Spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, August 2014. Conference Name: IEEE Transactions on Signal Processing.

[6] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223. PMLR, 2017.

[7] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.

[8] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *arXiv:1904.11955 [cs, stat]*, Nov 2019. arXiv: 1904.11955.

[9] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv:1901.08584 [cs, stat]*, May 2019. arXiv: 1901.08584.

[10] Devansh Arpit, Víctor Campos, and Yoshua Bengio. How to initialize your network? robust initialization for weightnorm & resnets. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[11] Francis Bach. On the Equivalence Between Kernel Quadrature Rules and Random Feature Expansions. *J. Mach. Learn. Res.*, 18(1):714–751, January 2017.

[12] Francis R. Bach. Exploring Large Feature Spaces with Hierarchical Multiple Kernel Learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 105–112. Curran Associates, Inc., 2009.

[13] Ricardo Baptista, Bamdad Hosseini, Nikola B. Kovachki, and Youssef Marzouk. Conditional sampling with monotone GANs: from generative models to likelihood-free inference. *arXiv:2006.06755*, 2023.

[14] Ricardo Baptista, Bamdad Hosseini, Nikola B Kovachki, Youssef M Marzouk, and Amir Sagiv. An approximation theory framework for measure-transport sampling algorithms. *arXiv:2302.13965*, 2023.

[15] Horace B Barlow et al. Possible principles underlying the transformation of sensory messages. *Sensory communication*, 1(01), 1961.

[16] Victor J. Barranca, Gregor Kovačič, Douglas Zhou, and David Cai. Improved compressive sensing of natural scenes using localized random sampling. *Scientific Reports*, 6(1):31976, Aug 2016.

[17] Friedrich G. Barth. Mechanics to pre-process information for the fine tuning of mechanoreceptors. *Journal of Comparative Physiology A*, 205(5):661–686, October 2019.

[18] Ronen Basri, Meirav Galun, Amnon Geifman, David Jacobs, Yoni Kasten, and Shira Kritchman. Frequency Bias in Neural Networks for Input of Non-Uniform Density. In *Proceedings of the 37th International Conference on Machine Learning*, pages 685–694. PMLR, November 2020. ISSN: 2640-3498.

[19] Ayanendranath Basu, Hiroyuki Shioya, and Chanseok Park. *Statistical inference: the minimum distance approach*. CRC press, 2011.

[20] Martin Bauer, Sarang Joshi, and Klas Modin. Diffeomorphic density matching by optimal information transport. *SIAM Journal on Imaging Sciences*, 8(3):1718–1751, 2015.

[21] M Faisal Beg, Michael I Miller, Alain Trouvé, and Laurent Younes. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *International journal of computer vision*, 61:139–157, 2005.

[22] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *arXiv:1206.5538 [cs]*, Apr 2014. arXiv: 1206.5538.

[23] David Beniaguev, Idan Segev, and Michael London. Single cortical neurons as deep artificial neural networks. *Neuron*, 109(17):2727–2739.e3, Sep 2021.

[24] Alain Berlinet and Christine Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Springer Science + Business Media, 2011.

[25] Alexandros Beskos, Mark Girolami, Shiwei Lan, Patrick E Farrell, and Andrew M Stuart. Geometric MCMC for infinite-dimensional inverse problems. *Journal of Computational Physics*, 335:327–351, 2017.

[26] Michael Betancourt, Simon Byrne, Sam Livingstone, and Mark Girolami. The geometric foundations of Hamiltonian Monte Carlo. *Bernoulli*, 23(4A):2257–2298, 2017.

[27] Felix Biggs, Antonin Schrab, and Arthur Gretton. Mmd-fuse: Learning and combining kernels for two-sample testing without data splitting. *Advances in Neural Information Processing Systems*, 36, 2024.

[28] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations (ICLR)*, 2018.

[29] Jeremiah Birrell, Paul Dupuis, Markos A Katsoulakis, Yannis Pantazis, and Luc Rey-Bellet. (f, γ)-divergences: Interpolating between f-divergences and integral probability metrics. *Journal of Machine Learning Research*, 23(39):1–70, 2022.

[30] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[31] Sergey Bobkov and Michel Ledoux. *One-dimensional empirical measures, order statistics, and Kantorovich transport distances*, volume 261. American Mathematical Society, 2019.

[32] Vincent Bonin, Mark H. Histed, Sergey Yurgenson, and R. Clay Reid. Local diversity and fine-scale organization of receptive fields in mouse visual cortex. *Journal of Neuroscience*, 31(50):18506–18521, Dec 2011.

[33] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51:22–45, 2015.

[34] Nicolas Bonnotte. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Université Paris Sud-Paris XI; Scuola normale superiore (Pise, Italie), 2013.

[35] Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. *arXiv:2002.02561 [cs, stat]*, Feb 2020. arXiv: 2002.02561.

[36] Blake Bordelon and Cengiz Pehlevan. Population codes enable learning from few examples by shaping inductive bias. *bioRxiv*, page 2021.03.30.437743, Apr 2021.

[37] Francois-Xavier Briol, Alessandro Barp, Andrew B Duncan, and Mark Girolami. Statistical inference for generative models with maximum mean discrepancy. *arXiv:1906.05944*, 2019.

[38] David S Broomhead and David Lowe. Radial basis functions, multi-variable functional interpolation and adaptive networks. Technical report, Royal Signals and Radar Establishment Malvern (United Kingdom), 1988.

[39] Joan Bruna and Stephane Mallat. Invariant Scattering Convolution Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, August 2013. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[40] Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *arXiv:2006.13198 [cond-mat, stat]*, Feb 2021. arXiv: 2006.13198.

[41] Claudio Canuto and Alfio Quarteroni. Approximation results for orthogonal polynomials in sobolev spaces. *Mathematics of Computation*, 38(157):67–86, 1982.

[42] Hanqun Cao, Cheng Tan, Zhangyang Gao, Guangyong Chen, Pheng-Ann Heng, and Stan Z Li. A survey on generative diffusion models. *arXiv:2209.02646*, 2022.

[43] Matteo Carandini, David J Heeger, and J Anthony Movshon. Linearity and normalization in simple cells of the macaque primary visual cortex. *Journal of Neuroscience*, 17(21):8621–8644, 1997.

[44] Sophie J. C. Caron, Vanessa Ruta, L. F. Abbott, and Richard Axel. Random convergence of olfactory inputs in the drosophila mushroom body. *Nature*, 497(74477447):113–117, May 2013.

[45] Matthew Chalk, Olivier Marre, and Gašper Tkačik. Toward a unified theory of efficient, predictive, and sparse coding. *Proceedings of the National Academy of Sciences*, 115(1):186–191, January 2018.

[46] Lin Chen and Sheng Xu. Deep neural tangent kernel and laplace kernel have the same rkhs. *arXiv:2009.10683 [cs, math, stat]*, Mar 2021. arXiv: 2009.10683.

[47] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[48] Xu Cheng-Long and Guo Ben-Yu. Hermite spectral and pseudospectral methods for nonlinear partial differential equation in multiple dimensions. *Computational & Applied Mathematics*, 22:167–193, 2003.

[49] E. J. Chichilnisky. A simple white noise analysis of neuronal light responses. *Network: Computation in Neural Systems*, 12(2):199–213, Feb 2001.

[50] Youngmin Cho and Lawrence Saul. Kernel methods for deep learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009.

[51] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv:1812.01718 [cs, stat]*, 9999. arXiv: 1812.01718.

[52] R. Clay Reid and Jose-Manuel Alonso. Specificity of monosynaptic connections from thalamus to visual cortex. *Nature*, 378(6554):281–284, Nov 1995.

[53] Jan Clemens and Bernhard Ronacher. Feature extraction and integration underlying perceptual decision making during courtship behavior. *Journal of Neuroscience*, 33(29):12136–12145, Jul 2013.

[54] SL Cotter, GO Roberts, AM Stuart, and D White. MCMC methods for functions: Modifying old algorithms to make them faster. *Statistical Science*, 28(3):424–446, 2013.

[55] Tiangang Cui, Kody JH Law, and Youssef M Marzouk. Dimension-independent likelihood-informed MCMC. *Journal of Computational Physics*, 304:109–137, 2016.

[56] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[57] Joel Dapello, Tiago Marques, Martin Schrimpf, Franziska Geiger, David Cox, and James J. DiCarlo. Simulating a primary visual cortex at the front of cnns improves robustness to image perturbations. *Advances in Neural Information Processing Systems*, 33, 2020.

[58] Lucas De Lara, Alberto González-Sanz, and Jean-Michel Loubes. Diffeomorphic registration using Sinkhorn divergences. *SIAM Journal on Imaging Sciences*, 16(1):250–279, 2023.

[59] Nabarun Deb, Promit Ghosal, and Bodhisattva Sen. Rates of estimation of optimal transport maps using plug-in estimators via barycentric projections. *Advances in Neural Information Processing Systems*, 34:29736–29753, 2021.

[60] Eustasio del Barrio, Alberto González Sanz, Jean-Michel Loubes, and Jonathan Niles-Weed. An improved central limit theorem and fast convergence rates for entropic transportation costs. *SIAM Journal on Mathematics of Data Science*, 5(3):639–669, 2023.

[61] Ishan Deshpande, Yuan-Ting Hu, Ruoyu Sun, Ayis Pyrros, Nasir Siddiqui, Sanmi Koyejo, Zhizhen Zhao, David Forsyth, and Alexander G Schwing. Max-sliced wasserstein distance and its use for gans. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10648–10656, 2019.

[62] Bradley H Dickerson, Jessica L Fox, and Simon Sponberg. Functional diversity from generic encoding in insect campaniform sensilla. *Current Opinion in Physiology*, 19:194–203, Feb 2021.

[63] Vincent Divol, Jonathan Niles-Weed, and Aram-Alexandre Pooladian. Optimal transport map estimation in general function spaces. *arXiv:2212.03722*, 2022.

[64] Alain Durmus and Éric Moulines. Nonasymptotic convergence analysis for the unadjusted langevin algorithm. *Annals of Applied Probability*, 27(3):1551–1587, 2017.

[65] Stanley Durrleman, Stéphanie Allassonnière, and Sarang Joshi. Sparse adaptive parameterization of variability in image ensembles. *International Journal of Computer Vision*, 101:161–183, 2013.

[66] Yonina C Eldar and Gitta Kutyniok, editors. *Compressed Sensing: Theory and Applications*. Cambridge University Press, 2012.

[67] Adrienne Fairhall. The receptive field is dead. long live the receptive field? *Current Opinion in Neurobiology*, 25:ix–xii, Apr 2014.

[68] Kion Fallah, Adam A. Willats, Ninghao Liu, and Christopher J. Rozell. Learning sparse codes from compressed representations with biologically plausible local wiring constraints. *bioRxiv*, 2020.

[69] Jean Feydy, Benjamin Charlier, François-Xavier Vialard, and Gabriel Peyré. Optimal transport for diffeomorphic registration. In *Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I 20*, pages 291–299. Springer, 2017.

[70] Jean Feydy and Alain Trouvé. Global divergences between measures: from Hausdorff distance to optimal transport. In *International Workshop on Shape in Medical Imaging*, pages 102–115. Springer, 2018.

[71] David Fitzpatrick. Seeing beyond the receptive field in primary visual cortex. *Current opinion in neurobiology*, 10(4):438–443, 2000.

[72] Rémi Flamary, Marco Cuturi, Nicolas Courty, and Alain Rakotomamonjy. Wasserstein discriminant analysis. *Machine Learning*, 107:1923–1945, 2018.

[73] Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing*. Birkhäuser Basel, 2013.

[74] Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure. *Probability theory and related fields*, 162(3):707–738, 2015.

[75] Charles W Fox and Stephen J Roberts. A tutorial on variational Bayesian inference. *Artificial intelligence review*, 38(2):85–95, 2012.

[76] J. L. Fox and T. L. Daniel. A neural basis for gyroscopic force measurement in the halteres of holorusia. *Journal of Comparative Physiology A*, 194(10):887–897, Oct 2008.

[77] Jessica L. Fox, Adrienne L. Fairhall, and Thomas L. Daniel. Encoding properties of haltere neurons enable motion feature detection in a biological gyroscope. *Proceedings of the National Academy of Sciences*, 107(8):3840–3845, Feb 2010.

[78] Jonathan Fritz, Shihab Shamma, Mounya Elhilali, and David Klein. Rapid task-related plasticity of spectrotemporal receptive fields in primary auditory cortex. *Nature Neuroscience*, 6:1216–1223, Nov 2003.

[79] Stefano Fusi, Earl K Miller, and Mattia Rigotti. Why neurons mix: high dimensionality for higher cognition. *Current Opinion in Neurobiology*, 37:66–74, Apr 2016.

[80] Alfred Galichon. *Optimal transport methods in economics*. Princeton University Press, 2018.

[81] Surya Ganguli and Haim Sompolinsky. Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis. *Annual Review of Neuroscience*, 35(1):485–508, 2012. PMID: 22483042.

[82] Alfredo Garbuno-Inigo, Franca Hoffmann, Wuchen Li, and Andrew M Stuart. Interacting langevin diffusions: Gradient structure and ensemble Kalman sampler. *SIAM Journal on Applied Dynamical Systems*, 19(1):412–441, 2020.

[83] François Gay-Balmaz, Darryl D Holm, David M Meier, Tudor S Ratiu, and François-Xavier Vialard. Invariant higher-order variational problems. *Communications in Mathematical Physics*, 309:413–458, 2012.

[84] Marc G. Genton. Classes of kernels for machine learning: A statistics perspective. *Journal of Machine Learning Research*, 2(Dec):299–312, 2001.

[85] Joan Glaunes, Alain Trouvé, and Laurent Younes. Diffeomorphic matching of distributions: A new approach for unlabelled point-sets and sub-manifolds matching. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II. IEEE, 2004.

[86] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, page 249–256. JMLR Workshop and Conference Proceedings, Mar 2010.

[87] Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Conference On Learning Theory*, pages 297–299. PMLR, 2018.

[88] Pieter M Goltstein, Guido T Meijer, and Cyriel MA Pennartz. Conditioning sharpens the spatial representation of rewarded stimuli in mouse primary visual cortex. *eLife*, 7:e37683, Sep 2018.

[89] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[90] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

[91] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[92] I. S. Gradshteyn and I. M. Ryzhik. *Table of integrals, series, and products*. Elsevier/Academic Press, Amsterdam, seventh edition, 2007. Translated from the Russian, Translation edited and with a preface by Alan Jeffrey and Daniel Zwillinger, With one CD-ROM (Windows, Macintosh and UNIX).

[93] Alexandre Gramfort, Gabriel Peyré, and Marco Cuturi. Fast optimal transport averaging of neuroimaging data. In *Information Processing in Medical Imaging: 24th International Conference, IPMI 2015, Sabhal Mor Ostaig, Isle of Skye, UK, June 28-July 3, 2015, Proceedings 24*, pages 261–272. Springer, 2015.

[94] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2018.

[95] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.

[96] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of Wasserstein GANs. In *International Conference on Neural Information Processing Systems*, 2017.

[97] Martin Hairer, Andrew M Stuart, Sebastian J Vollmer, et al. Spectral gaps for a Metropolis–Hastings algorithm in infinite dimensions. *The Annals of Applied Probability*, 24(6):2455–2490, 2014.

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

[99] Kameron Decker Harris. Additive function approximation in the brain. *arXiv:1909.02603 [cs, q-bio, stat]*, Sep 2019. arXiv: 1909.02603.

[100] Abolfazl Hashemi, Hayden Schaeffer, Robert Shi, Ufuk Topcu, Giang Tran, and Rachel Ward. Generalization bounds for sparse random feature expansions. *arXiv:2103.03191 [cs, math, stat]*, Aug 2021. arXiv: 2103.03191.

[101] Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258, Jul 2017.

[102] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv:1512.03385 [cs]*, Dec 2015. arXiv: 1512.03385.

[103] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, page 1026–1034, Dec 2015.

[104] Bamdad Hosseini. Two Metropolis–Hastings algorithms for posterior measures with non-Gaussian priors in infinite dimensions. *SIAM/ASA Journal on Uncertainty Quantification*, 7(4):1185–1223, 2019.

[105] Bamdad Hosseini and James E Johndrow. Spectral gaps and error estimates for infinite-dimensional metropolis–hastings with non-gaussian priors. *The Annals of Applied Probability*, 33(3):1827–1873, 2023.

[106] Jian Huang, Yuling Jiao, Zhen Li, Shiao Liu, Yang Wang, and Yunfei Yang. An error analysis of generative adversarial networks for learning distributions. *Journal of machine learning research*, 23(116):1–43, 2022.

[107] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591, Oct 1959.

[108] Jan-Christian Hütter and Philippe Rigollet. Minimax estimation of smooth optimal transport maps. *The Annals of Statistics*, 49(2), 2021.

[109] B. Igelnik and Y. H. Pao. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. *IEEE transactions on neural networks*, 6(6):1320–1329, 1995.

[110] Bernd Illing, Wulfram Gerstner, and Johanni Brea. Biologically plausible deep learning - but how far can we go with shallow networks? *Neural Networks*, 118:90–101, Oct 2019.

[111] Nicholas J Irons, Meyer Scetbon, Soumik Pal, and Zaid Harchaoui. Triangular flows for generative modeling: Statistical consistency, smoothness classes, and fast rates. In *International Conference on Artificial Intelligence and Statistics*, pages 10161–10195. PMLR, 2022.

[112] Isao Ishikawa, Takeshi Teshima, Koichi Tojo, Kenta Oono, Masahiro Ikeda, and Masashi Sugiyama. Universal approximation property of invertible neural networks. *Journal of Machine Learning Research*, 24(287):1–68, 2023.

[113] Jörn-Henrik Jacobsen, Jan van Gemert, Zhongyu Lou, and Arnold W. M. Smeulders. Structured receptive fields in cnns. *arXiv:1605.02971 [cs]*, May 2016. arXiv: 1605.02971.

[114] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[115] Tony Jebara. *Machine learning: discriminative and generative*, volume 755. Springer Science & Business Media, 2012.

[116] J. P. Jones and L. A. Palmer. An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, 58(6):1233–1258, Dec 1987.

[117] Sarang C Joshi and Michael I Miller. Landmark matching via large deformation diffeomorphisms. *IEEE transactions on image processing*, 9(8):1357–1370, 2000.

[118] Hachem Kadri, Emmanuel Duflos, Philippe Preux, Stéphane Canu, Alain Rakotomamonjy, and Julien Audiffren. Operator-valued kernels for learning from functional response data. *The Journal of Machine Learning Research*, 17(1):613–666, 2016.

[119] Alexander Kell, Daniel Yamins, Sam Norman-Haignere, Darren Seibert, Ha Hong, Jim DiCarlo, and Josh McDermott. Computational similarities between visual and auditory cortex studied with convolutional neural networks, fmri, and electrophysiology. *Journal of Vision*, 15(12):1093–1093, Sep 2015.

[120] M Kleiner, D Brainard, and D Pelli. What's new in psychtoolbox-3? In *Perception - ECVP Abstract Supplement. European Conference on Visual Perception (ECVP-2007), August 27-31, Arezzo, Italy*, 2007.

[121] E. I. Knudsen and M. Konishi. Center-surround organization of auditory receptive fields in the owl. *Science*, 202(4369):778–780, Nov 1978.

[122] Sue Ann Koay, Adam S. Charles, Stephan Y. Thiberge, Carlos D. Brody, and David W. Tank. Sequential and efficient neural-population coding of complex task information. *bioRxiv*, 2021.

[123] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

[124] Alexander V Kolesnikov. On sobolev regularity of mass transport and transportation inequalities. *Theory of Probability & Its Applications*, 57(2):243–264, 2013.

[125] D. D. Kosambi. Statistics in function space. *The Journal of the Indian Mathematical Society. New Series*, 7:76–88, 1943.

[126] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv:1404.5997 [cs]*, April 2014. arXiv: 1404.5997.

[127] Daniel Kuhn, Peyman Mohajerin Esfahani, Viet Anh Nguyen, and Soroosh Shafieezadeh-Abadeh. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations research & management science in the age of analytics*, pages 130–166. Informs, 2019.

[128] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[129] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. *Advances in neural information processing systems*, 30, 2017.

[130] Qianxiao Li, Ting Lin, and Zuowei Shen. Deep learning via dynamical systems: An approximation perspective. *Journal of the European Mathematical Society*, 25(5):1671–1709, 2022.

[131] Tengyuan Liang. How well generative adversarial networks learn distributions. *Journal of Machine Learning Research*, 22(228):1–41, 2021.

[132] Grace W. Lindsay. Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future. *Journal of Cognitive Neuroscience*, pages 1–15, February 2020.

[133] Ashok Litwin-Kumar, Kameron Decker Harris, Richard Axel, Haim Sompolinsky, and L.F. Abbott. Optimal degrees of synaptic connectivity. *Neuron*, 93(5):1153–1164.e7, Mar 2017.

[134] Fanghui Liu, Xiaolin Huang, Yudong Chen, and Johan A. K. Suykens. Random features for kernel approximation: A survey in algorithms, theory, and beyond. *arXiv:2004.11154 [cs, stat]*, Apr 2020. arXiv: 2004.11154.

[135] Stéphane Mallat. Group Invariant Scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

[136] Tudor Manole, Sivaraman Balakrishnan, and Larry Wasserman. Minimax confidence intervals for the sliced wasserstein distance. *Electronic Journal of Statistics*, 16(1):2252–2345, 2022.

[137] D. Marr, E. Hildreth, and Sydney Brenner. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217, February 1980. Publisher: Royal Society.

[138] J.-B. Martens. The Hermite transform-theory. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(9):1595–1606, September 1990. Conference Name: IEEE Transactions on Acoustics, Speech, and Signal Processing.

[139] James Martens et al. Deep learning via hessian-free optimization. In *Icml*, volume 27, pages 735–742, 2010.

[140] Youssef Marzouk, Tarek Moselhy, Matthew Parno, and Alessio Spantini. Sampling via measure transport: An introduction. *Handbook of Uncertainty Quantification*, pages 1–41, 2016.

[141] Youssef Marzouk, Zhi Ren, Sven Wang, and Jakob Zech. Distribution learning via neural differential equations: a nonparametric statistical perspective. *arXiv preprint arXiv:2309.01043*, 2023.

[142] Robert Mattheij and Jaap Molenaar. *Ordinary differential equations in theory and practice*. SIAM, 2002.

[143] César Lincoln C. Mattos, Zhenwen Dai, Andreas Damianou, Jeremy Forth, Guilherme A. Barreto, and Neil D. Lawrence. Recurrent gaussian processes. *arXiv:1511.06644 [cs, stat]*, Feb 2016. arXiv: 1511.06644.

[144] Theodor Misiakiewicz and Song Mei. Learning with convolution and pooling operations in kernel methods, June 2022.

[145] Thomas L. Mohren, Thomas L. Daniel, Steven L. Brunton, and Bingni W. Brunton. Neural-inspired sensors enable sparse, efficient classification of spatiotemporal data. *Proceedings of the National Academy of Sciences*, 115(42):10564–10569, Oct 2018.

[146] Eduardo Fernandes Montesuma and Fred Maurice Ngole Mboula. Wasserstein barycenter for multi-source domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16785–16793, 2021.

[147] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

[148] Radford M. Neal. *Priors for Infinite Networks*, pages 29–53. Springer New York, New York, NY, 1996.

[149] Andrew Ng and Michael Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, 14, 2001.

[150] Jonathan Niles-Weed and Philippe Rigollet. Estimation of wasserstein distances in the spiked transport model. *Bernoulli*, 28(4):2663–2688, 2022.

[151] Erkki Oja. Principal components, minor components, and linear neural networks. *Neural Networks*, 5(6):927–935, November 1992.

[152] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization.

[153] Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37(23):3311–3325, December 1997.

[154] Derek Onken, S Wu Fung, Xingjian Li, and Lars Ruthotto. OT-flow: Fast and accurate continuous normalizing flows via optimal transport. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 2021.

[155] Srdjan Ostojic and Nicolas Brunel. From Spiking Neuron Models to Linear-Nonlinear Models. *PLOS Computational Biology*, 7(1):e1001056, January 2011. Publisher: Public Library of Science.

[156] Houman Owhadi. Do ideas have shape? idea registration as the continuous limit of artificial neural networks. *Physica D: Nonlinear Phenomena*, 444:133592, 2023.

[157] Houman Owhadi and Clint Scovel. *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization: From a Game Theoretic Approach to Numerical Approximation and Algorithm Design*, volume 35. Cambridge University Press, 2019.

[158] Marius Pachitariu, Carsen Stringer, Mario Dipoppa, Sylvia Schröder, L Federico Rossi, Henry Dalgleish, Matteo Carandini, and Kenneth D Harris. Suite2p: beyond 10,000 neurons with standard two-photon microscopy. *BioRxiv*, 2017.

[159] Biraj Pandey, Bamdad Hosseini, Pau Batlle, and Houman Owhadi. Diffeomorphic measure matching with kernels for generative modeling, 2024.

[160] Biraj Pandey, Marius Pachitariu, Bingni W Brunton, and Kameron Decker Harris. Structured random receptive fields enable informative sensory encodings. *PLoS Computational Biology*, 18(10):e1010484, 2022.

[161] Liam Paninski. Convergence properties of some spike-triggered analysis techniques. In *Network: Computation in Neural Systems*, page 2003, 2003.

[162] George Papamakarios, Eric T Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.

[163] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[164] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[165] Leandro Pardo. *Statistical inference based on divergence measures*. CRC press, 2018.

[166] Mijung Park and Jonathan W Pillow. Receptive field inference with localized priors. *PLoS computational biology*, 7(10):e1002219, 2011.

[167] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[168] François-Pierre Paty and Marco Cuturi. Subspace robust wasserstein distances. In *International conference on machine learning*, pages 5072–5081. PMLR, 2019.

[169] Cengiz Pehlevan, Anirvan M. Sengupta, and Dmitri B. Chklovskii. Why Do Similarity Matching Objectives Lead to Hebbian/Anti-Hebbian Networks? *Neural Computation*, 30(1):84–124, January 2018.

[170] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[171] Silvia L. Pintea, Nergis Tomen, Stanley F. Goes, Marco Loog, and Jan C. van Gemert. Resolution learning in deep convolutional networks using scale-space theory. *arXiv:2106.03412 [cs]*, Jun 2021. arXiv: 2106.03412.

[172] Aram-Alexandre Pooladian, Vincent Divol, and Jonathan Niles-Weed. Minimax estimation of discontinuous optimal transport maps: The semi-discrete case. In *40th International Conference on Machine Learning (ICML 2023)*, 2023.

[173] Brandon Pratt, Tanvi Deora, Thomas Mohren, and Thomas Daniel. Neural evidence supports a dual sensory-motor role for insect wings. *Proceedings of the Royal Society B: Biological Sciences*, 284(1862):20170969, Sep 2017.

[174] J. Andrew Pruszynski and Roland S. Johansson. Edge-orientation processing in first-order tactile neurons. *Nature Neuroscience*, 17(10):1404–1409, Oct 2014.

[175] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein barycenter and its application to texture mixing. In *Scale Space and Variational Methods in Computer Vision: Third International Conference, SSVM 2011, Ein-Gedi, Israel, May 29–June 2, 2011, Revised Selected Papers 3*, pages 435–446. Springer, 2012.

[176] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[177] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1177–1184. Curran Associates, Inc., 2008.

[178] Ali Rahimi and Benjamin Recht. Uniform approximation of functions with random bases. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, page 555–561. IEEE, Sep 2008.

[179] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[180] Matteo Raviola. Training kernel neural ODEs with optimal control and Riemannian optimization, 2022. (Masters thesis) Politecnico di Torino.

[181] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, 2015.

[182] A. Reznikov and E. B. Saff. The Covering Radius of Randomly Distributed Points on a Manifold. *International Mathematics Research Notices*, 2016(19):6065–6094, 12 2015.

[183] Dario L. Ringach. Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *Journal of Neurophysiology*, 88(1):455–463, Jul 2002.

[184] Dario L. Ringach. Haphazard Wiring of Simple Receptive Fields and Orientation Columns in Visual Cortex. *Journal of Neurophysiology*, 92(1):468–476, July 2004.

[185] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.

[186] R. T. Rockafellar and R. J. B. Wets. *Variational Analysis*. Springer Berlin, Heidelberg, 2005.

[187] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 19590501.

[188] Juha Rusanen, Roman Frolov, Matti Weckström, Michiyo Kinoshita, and Kentaro Arikawa. Non-linear amplification of graded voltage signals in the first-order visual interneurons of the butterfly papilio xuthus. *Journal of Experimental Biology*, 221(12), Jun 2018.

[189] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, December 2015.

[190] Lars Ruthotto and Eldad Haber. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, 62, 2020.

[191] Hsiao S. Central mechanisms of tactile shape perception. *Current opinion in neurobiology*, 18(4), Aug 2008.

[192] H. M. Sakai and K. Naka. Signal transmission in the catfish retina. v. sensitivity and circuit. *Journal of Neurophysiology*, 58(6):1329–1350, Dec 1987.

[193] Filippo Santambrogio. *Optimal transport for applied mathematicians*. Springer, 2015.

[194] Shreya Saxena and John P Cunningham. Towards the neural population doctrine. *Current Opinion in Neurobiology*, 55:103–111, Apr 2019.

[195] Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, et al. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943, 2019.

[196] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[197] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[198] Charles Sherrington. *The Integrative Action of the Nervous System*. Cambridge University Press, 1907.

[199] Shashank Singh and Barnabás Póczos. Minimax distribution estimation in wasserstein distance. *arXiv preprint arXiv:1802.08855*, 2018.

[200] Shashank Singh, Ananya Uppal, Boyue Li, Chun-Liang Li, Manzil Zaheer, and Barnabás Póczos. Nonparametric density estimation under adversarial losses. *Advances in Neural Information Processing Systems*, 31, 2018.

[201] Aman Sinha and John C Duchi. Learning kernels with random features. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[202] Nicholas James Sofroniew, Daniel Flickinger, Jonathan King, and Karel Svoboda. A large field of view two-photon mesoscope with subcellular resolution for in vivo imaging. *Elife*, 5:e14472, 2016.

[203] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

[204] Bharath Sriperumbudur. On the optimal estimation of probability measures in weak and strong topologies. *Bernoulli*, 22(3):1839–1893, 2016.

[205] Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Gert RG Lanckriet. On the empirical estimation of integral probability metrics. 2012.

[206] Gilbert Strang. The Discrete Cosine Transform. *SIAM Review*, 41(1):135–147, January 1999.

[207] Nicholas James Strausfeld. *Arthropod Brains: Evolution, Functional Elegance, and Historical Significance*. Harvard University Press, 2012.

[208] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571(7765):361–365, 2019.

[209] Andrew M Stuart. Inverse problems: a Bayesian perspective. *Acta numerica*, 19:451–559, 2010.

[210] Zhengyu Su, Yalin Wang, Rui Shi, Wei Zeng, Jian Sun, Feng Luo, and Xianfeng Gu. Optimal mass transport for shape matching and comparison. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2246–2259, 2015.

[211] Larry W. Swanson. *Brain architecture: Understanding the basic plan*, volume xv. Oxford University Press, New York, NY, US, 2003.

[212] Behrooz Tahmasebi and Stefanie Jegelka. Sample complexity bounds for estimating probability divergences under invariances. *arXiv preprint arXiv:2311.02868*, 2023.

[213] Rong Tang and Yun Yang. Minimax rate of distribution estimation on unknown submanifolds under adversarial losses. *The Annals of Statistics*, 51(3):1282–1308, 2023.

[214] Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

[215] Ilya Tolstikhin, Bharath K Sriperumbudur, Krikamol Mu, et al. Minimax estimation of kernel mean embeddings. *Journal of Machine Learning Research*, 18(86):1–47, 2017.

[216] Lloyd N Trefethen. *Approximation theory and approximation practice, extended edition*. SIAM, 2019.

[217] Alain Trouvé and François-Xavier Vialard. Shape splines and stochastic shape evolutions: A second order point of view. *Quarterly of Applied Mathematics*, pages 219–251, 2012.

[218] Rui Tuo and CF Jeff Wu. A theoretical framework for calibration in computer models: Parametrization, estimation and convergence properties. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):767–795, 2016.

[219] Cédric Villani. *Optimal transport: Old and new*. Springer, 2009.

[220] Cédric Villani. *Optimal transport: old and new*. Springer, 2009.

[221] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W.

Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[222] Grace Wahba. *Spline Models for Observational Data*. SIAM, September 1990.

[223] Christian Walder and Bernhard Schölkopf. Diffeomorphic dimensionality reduction. *Advances in Neural Information Processing Systems*, 21, 2008.

[224] Sven Wang and Youssef Marzouk. On minimax density estimation via measure transport. *arXiv:2207.10231*, 2022.

[225] Zheyu Oliver Wang, Ricardo Baptista, Youssef Marzouk, Lars Ruthotto, and Deepanshu Verma. Efficient neural network approaches for conditional optimal transport with applications in bayesian inference. *arXiv preprint arXiv:2310.16975*, 2023.

[226] Alison I. Weber and Jonathan W. Pillow. Capturing the Dynamical Repertoire of Single Neurons with Generalized Linear Models. *Neural Computation*, 29(12):3260–3289, December 2017.

[227] Jonathan Weed and Quentin Berthet. Estimation of smooth densities in wasserstein distance. In *conference on Learning Theory*, pages 3118–3119. PMLR, 2019.

[228] Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

[229] Christopher K. I. Williams. Computation with infinite neural networks. *Neural Computation*, 10(5):1203–1216, Jul 1998.

[230] Marjorie Xie, Samuel Muscinelli, Kameron Decker Harris, and Ashok Litwin-Kumar. Understanding the role of sparseness in cerebellar granule cell representations. In *Computational and Systems Neuroscience (Cosyne)*. 2021.

[231] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. A temporal kernel approach for deep learning with continuous-time information. *arXiv:2103.15213 [cs]*, Mar 2021. arXiv: 2103.15213.

[232] Daniel L. K. Yamins, Ha Hong, Charles F. Cadieu, Ethan A. Solomon, Darren Seibert, and James J. DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014.

[233] Alexandra M. Yarger and Jessica L. Fox. Dipteran halteres: Perspectives on function and integration for a unique sensory organ. *Integrative and Comparative Biology*, 56(5):865–876, Nov 2016.

[234] Laurent Younes. *Shapes and diffeomorphisms*. Springer, 2010.

[235] Laurent Younes. Constrained diffeomorphic shape evolution. *Foundations of Computational Mathematics*, 12:295–325, 2012.

[236] Laurent Younes. Diffeomorphic learning. (arXiv:1806.01240), Oct 2019. arXiv:1806.01240 [cs, stat].

[237] Rafael Yuste. From the neuron doctrine to neural networks. *Nature Reviews Neuroscience*, 16(8):487–497, Aug 2015.

[238] Anthony M. Zador. A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature Communications*, 10(1):3770, Dec 2019.

[239] Jakob Zech and Youssef Marzouk. Sparse approximation of triangular transports, part i: The finite-dimensional case. *Constructive Approximation*, 55(3):919–986, 2022.

[240] Jakob Zech and Youssef Marzouk. Sparse approximation of triangular transports, part ii: The infinite-dimensional case. *Constructive Approximation*, 55(3):987–1036, 2022.

[241] Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

[242] Kai Zhang, Ivor W. Tsang, and James T. Kwok. Improved nyström low-rank approximation and error analysis. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, page 1232–1239, Helsinki, Finland, 2008. ACM Press.

[243] Charlie W. Zhao, Mark J. Daley, and J. Andrew Pruszynski. Neural network models of the tactile system develop first-order units with spatially complex receptive fields. *PLOS ONE*, 13(6):e0199196, Jun 2018.

[244] Yubo Zhuang, Xiaohui Chen, and Yun Yang. Wasserstein k-means for clustering probability distributions. *Advances in Neural Information Processing Systems*, 35:11382–11395, 2022.

[245] Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, Oct 2003.

Appendix A

DIFFEOMORPHIC MEASURE MATCHING WITH KERNELS FOR GENERATIVE MODELING

A.1 Numerical results from autonomous KODE

Figure A.1 shows the result from autonomous KODE on two-dimensional benchmarks.

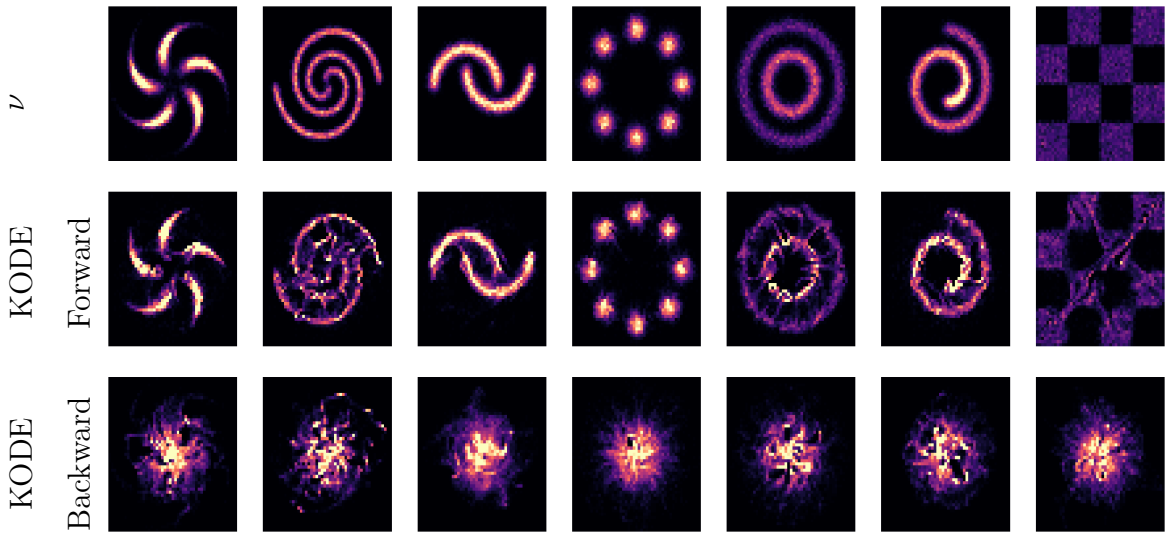

Figure A.1: **Transport experiments on two dimensional benchmarks using autonomous KODE.** (Top row) The empirical samples from complex measure ν. (Middle row) Samples generated after transporting isotropic gaussian measure η. (Bottom row) Samples generated from the backward flow of KODE on ν.

A.2 MMD comparison with different kernels

We compute MMD with four additional kernels: RBF ($\sigma = 1$), RBF-med ($\sigma =$ median distance), RBF-med-0.25 ($\sigma = 0.25 *$ median distance), and polynomial kernel (degree $= 2$). Table A.1 and

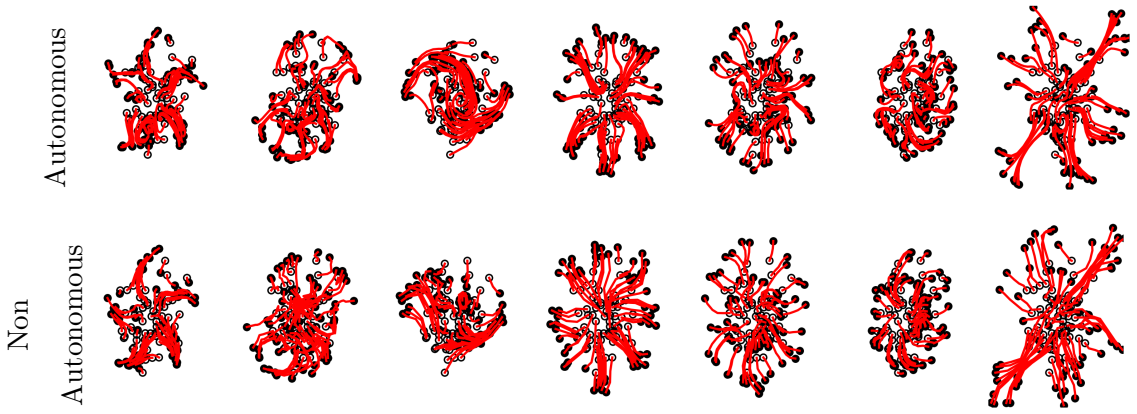

Figure A.2: **Trajectory of transport for 2-D examples using KODE going from reference η (open circles) to final location (closed circles)**. (Top row) Forward flow of autonomous KODE. (Bottom row) Forward flow of non-autonomous KODE.

Table A.2 shows the results.

A.3 Quantile-Quantile plot for 2D benchmarks

We compare the marginal distributions of the original reference measure η with the predictions from the backward flow of the KODE model using a quantile-quantile (Q-Q) plot. Figure A.3 and Figure A.4 show the results for autonomous and non-autonomous KODE models, respectively. For both models, there is a good match between the model quantiles and η quantiles along each dimension.

A.4 Hyperparameters

Table A.3 and Table A.4 outline the hyperparameters used for each method to generate the results in both main and supplementary figures.

A.5 Additional numerical results for high-dimensional benchmarks

Here we show the marginals for the remaining high-dimensional benchmarks. Figure A.5–Figure A.7 show the results for non-autonomous KODE models. Figure A.8–Figure A.12 show the results for

Table A.1: **Normalized MMD on 2D benchmark examples with different MMD kernels.**

Dataset	Model	RBF (↓)	RBF-med (↓)	RBF-med-0.25 (↓)	Poly2 (↓)
Pinwheel	KODE (non-auto)	2.8e-03	1.9e-03	4.0e-03	1.8e-03
	KODE (auto)	2.6e-03	**1.5e-03**	3.4e-03	**1.4e-03**
	OT-FLOW	**1.5e-03**	2.1e-03	**1.8e-03**	2.0e-03
2spirals	KODE (non-auto)	7.6e-03	**4.1e-03**	1.0e-02	4.1e-03
	KODE (auto)	**5.5e-03**	7.0e-03	7.1e-03	**2.3e-03**
	OT-FLOW	5.8e-03	7.4e-03	**6.2e-03**	1.1e-02
moons	KODE (non-auto)	7.5e-03	**2.9e-03**	9.8e-03	**2.6e-03**
	KODE (auto)	**2.8e-03**	5.2e-04	**4.1e-03**	2.9e-04
	OT-FLOW	4.5e-03	5.1e-03	5.7e-03	4.2e-03
8gaussians	KODE (non-auto)	1.1e-03	6.0e-04	1.1e-03	1.2e-03
	KODE (auto)	1.0e-03	**3.5e-04**	9.5e-04	**8.6e-06**
	OT-FLOW	**4.1e-04**	4.9e-04	**4.0e-04**	4.6e-04
circles	KODE (non-auto)	**3.0e-03**	3.2e-03	**3.4e-03**	**3.5e-04**
	KODE (auto)	4.3e-03	**2.6e-03**	4.5e-03	1.5e-03
	OT-FLOW	7.1e-03	9.4e-03	7.2e-03	1.2e-02
swissroll	KODE (non-auto)	4.3e-03	6.2e-03	5.3e-03	1.5e-03
	KODE (auto)	3.5e-03	6.5e-03	4.6e-03	**3.9e-04**
	OT-FLOW	**2.6e-03**	**3.2e-03**	**3.0e-03**	3.2e-03
checkerboard	KODE (non-auto)	2.9e-03	9.8e-04	2.3e-03	1.8e-03
	KODE (auto)	**1.0e-03**	**7.9e-04**	**8.0e-04**	**6.3e-04**
	OT-FLOW	2.2e-03	3.0e-03	1.9e-03	5.7e-03

autonomous KODE models.

Table A.2: **Normalized MMD on high-dimensional benchmark examples with different MMD kernels.**

Dataset	Model	RBF (↓)	RBF-med (↓)	RBF-med-0.25 (↓)	Poly2 (↓)
`Power` (d=6)	KODE (non-auto)	6.1e-03	2.1e-01	4.9e-03	1.0e+00
	KODE (auto)	**1.1e-03**	3.4e-02	**9.5e-04**	3.2e-01
	OT-FLOW	2.5e-03	**2.1e-02**	2.0e-03	**3.6e-02**
`GAS` (d=8)	KODE (non-auto)	1.5e-02	2.5e-02	6.1e-03	3.7e-02
	KODE (auto)	**4.7e-03**	5.3e-03	**2.3e-03**	2.7e-02
	OT-FLOW	5.3e-03	**3.6e-03**	3.3e-03	**8.7e-03**
`HEPMASS` (d=21)	KODE (non-auto)	2.0e+00	8.3e-01	5.1e-01	5.8e-01
	KODE (auto)	3.0e+00	5.9e-01	7.0e-01	1.8e-01
	OT-FLOW	**9.3e-01**	**1.2e-02**	**2.6e-02**	**3.6e-03**
`MINIBOONE` (d=43)	KODE (non-auto)	1.0e+00	5.1e-01	3.8e-01	3.3e-01
	KODE (auto)	1.1e+00	4.0e-01	3.0e-01	3.9e-01
	OT-FLOW	1.0e+00	**4.3e-02**	**3.7e-02**	**2.3e-02**
`BSDS300` (d=63)	KODE (non-auto)	3.4e-01	2.7e+00	2.8e-02	1.1e+00
	KODE (auto)	4.5e-02	3.9e-01	1.8e-02	2.2e-01
	OT-FLOW	**8.2e-03**	**1.6e-01**	**1.0e-02**	**1.5e-01**

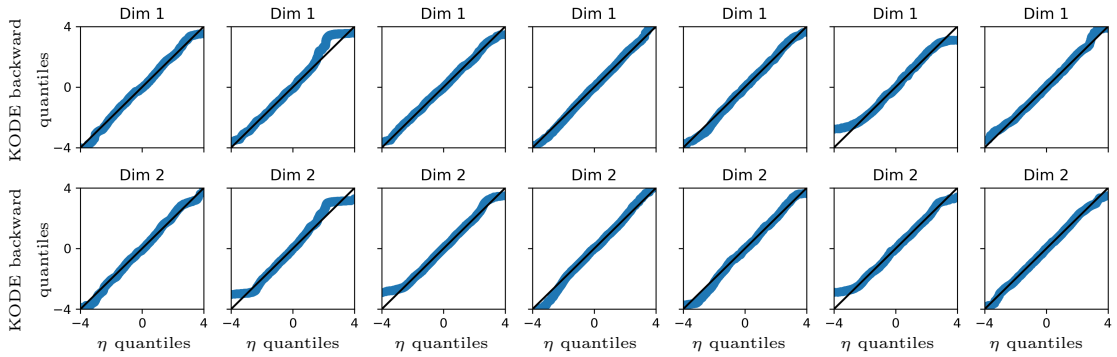

Figure A.3: **Quantile-Quantile (QQ) plots for two-dimensional toy examples using autonomous KODE.** We compare the quantiles of the gaussian measure η with the predictions from the backward flow of the model. From left to right, each column shows the quantiles for pinwheel, 2spirals, moons, 8gaussians, circles, swissroll, and checkerboard respectively. Each row corresponds to a dimension.

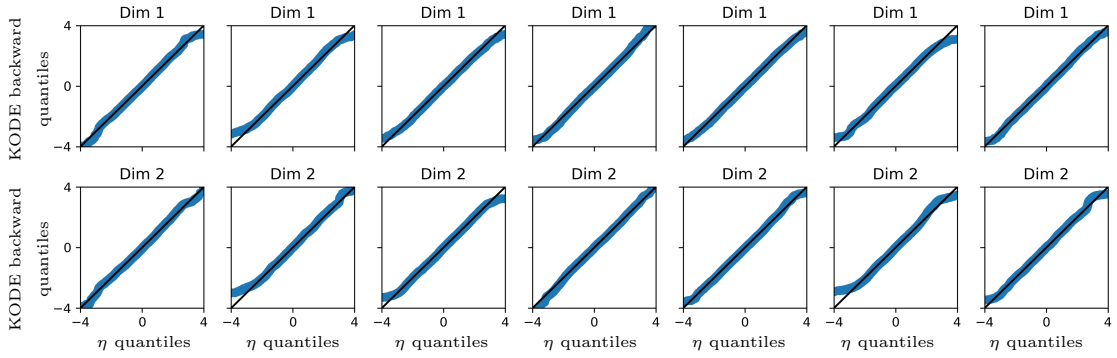

Figure A.4: **Quantile-Quantile (QQ) plots for two-dimensional toy examples using non-autonomous KODE.** We compare the quantiles of the gaussian measure η with the predictions from the backward flow of the model. From left to right, each column shows the quantiles for pinwheel, 2spirals, moons, 8gaussians, circles, swissroll, and checkerboard respectively. Each row corresponds to a dimension.

Table A.3: **Hyperparameters for autonomous Kernel ODE for benchmark examples.** For all examples, we use the `EulerHeun` solver with $n = 10$ integration steps. We use the Radial Basis Function (RBF) kernel as the model kernel and the Laplace kernel for the MMD loss. For the learning rate, we use an exponential decay rate scheduler starting at $lr = 0.1$ and decreasing exponentially by 0.5 every 100 iterations.

	Time indep. KODE parameters				**MMD Loss parameters**	
Dataset	#landmarks	length scale	batch size	#epochs	length scale	λ_1
Pinwheel	400	0.15	5000	601	0.47	1e-6
2spirals	200	0.15	5000	301	0.47	1e-7
Moons	100	0.29	5000	301	0.44	1e-7
8gaussians	400	0.18	5000	601	0.55	1e-6
Circles	400	0.17	5000	301	0.52	1e-6
Swissroll	400	0.15	5000	301	0.45	1e-6
Checkerboard	500	0.21	10000	601	0.64	1e-6
`POWER`	1000	0.68	2048	5	0.68	1e-8
`GAS`	1000	0.86	2048	5	0.86	1e-10
`HEPMASS`	5000	0.9	1024	20	1.8	1e-10
`MINIBOONE`	1500	1.9	1024	20	3.8	1e-10
`BSDS300`	1000	2.95	512	20	5.9	1e-10
`Lotka-Volterra`	5000	0.71	2048	401	1.43	1e-11

Table A.4: **Hyperparameters for non-autonomous Kernel ODE for benchmark examples.** For all examples within each discrete steps, we use the `EulerHeun` solver with $n = 10$ integration steps. We use the Radial Basis Function (RBF) kernel as the model kernel and the Laplace kernel for the MMD loss. For the learning rate, we use an exponential decay rate scheduler starting at $lr = 0.1$ and decreasing exponentially by 0.5 every 100 iterations.

	Time dep. KODE parameters					**MMD Loss parameters**		
Dataset	#landmarks	Len. scale	# discrete steps	Batch size	#epochs	length scale	λ_1	λ_2
Pinwheel	100	0.31	5	5000	501	0.47	1e-6	1e-6
2spirals	100	0.31	5	5000	501	0.47	1e-9	1e-8
Moons	150	0.30	3	5000	501	0.45	1e-9	1e-6
8gaussians	150	0.36	3	5000	701	0.54	1e-6	1e-5
Circles	100	0.35	5	5000	501	0.52	1e-6	1e-7
Swissroll	100	0.30	5	5000	501	0.45	1e-7	1e-6
Checkerboard	150	0.21	4	10000	1001	0.64	1e-8	1e-8
`POWER`	1000	0.74	2	2048	11	0.74	1e-11	1e-11
`GAS`	2000	0.96	4	2048	11	0.96	1e-11	1e-11
`HEPMASS`	2000	1.16	2	1024	20	2.32	1e-11	1e-11
`MINIBOONE`	1000	1.82	2	1024	20	3.63	1e-11	1e-11
`BSDS300`	2000	4.13	2	512	7	8.26	1e-11	1e-11
`MNIST`	3500	0.5	2	1024	101	1.04	1e-11	1e-11

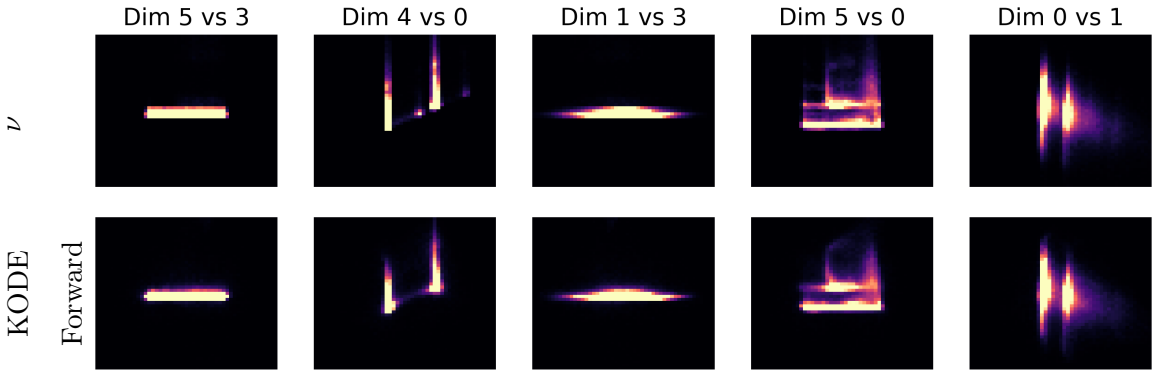

Figure A.5: **Transport experiments on `POWER` benchmark using non-autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

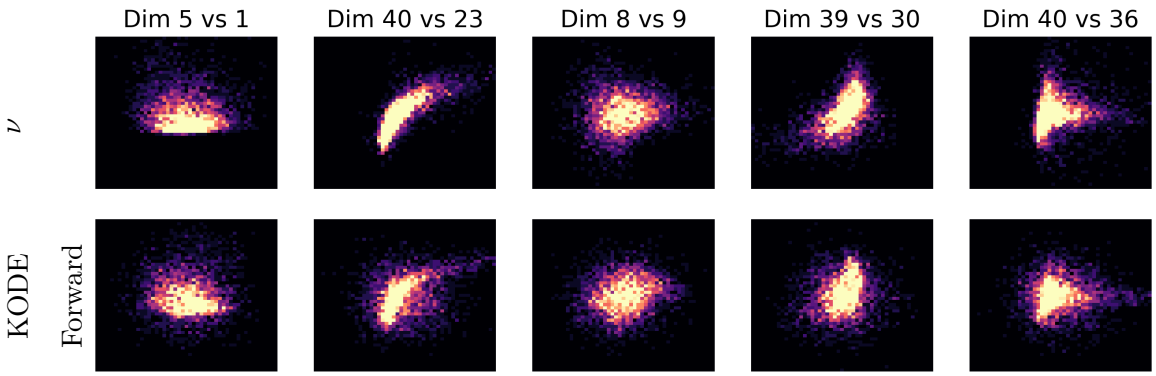

Figure A.6: **Transport experiments on `MINIBOONE` benchmark using non-autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

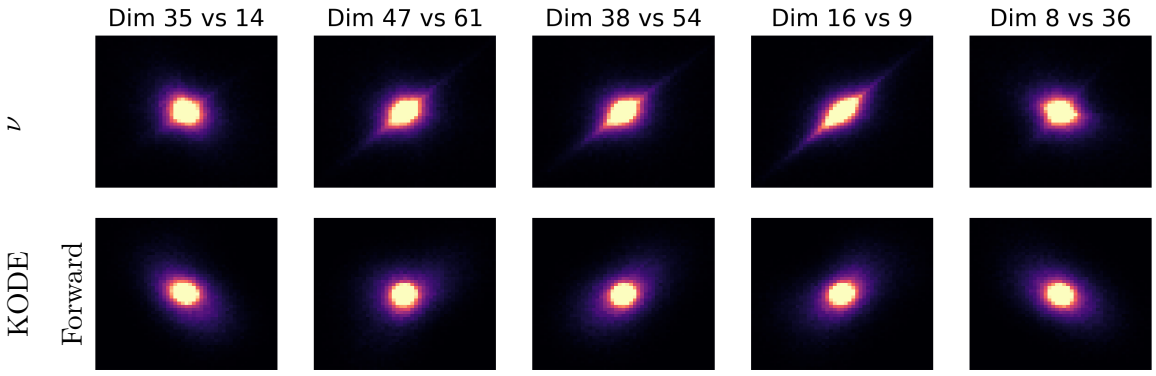

Figure A.7: **Transport experiments on `BSDS300` benchmark using non-autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

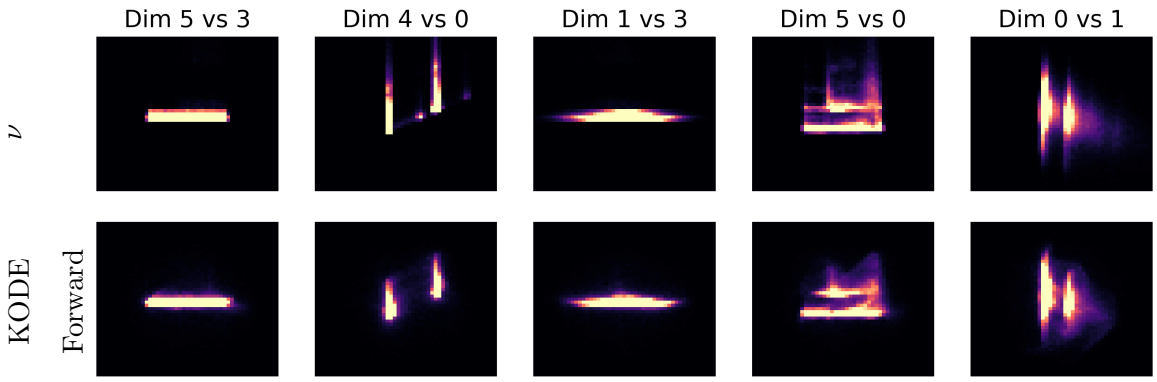

Figure A.8: **Transport experiments on `POWER` benchmark using autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

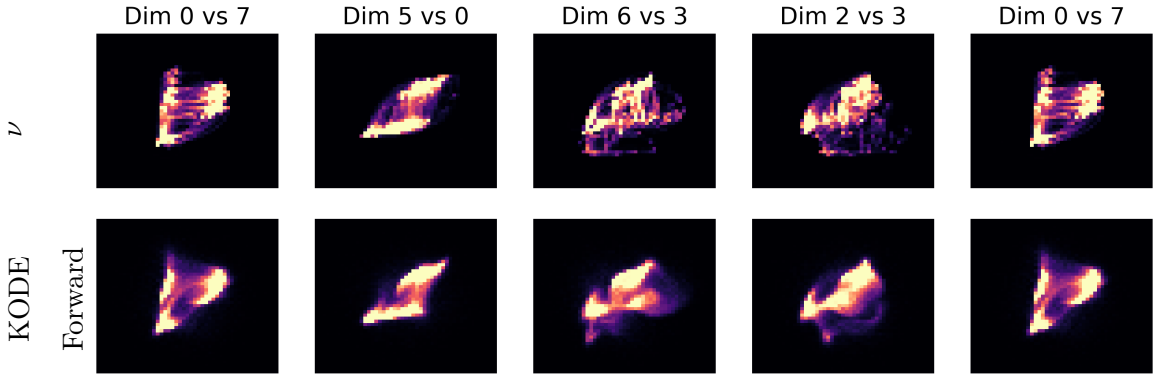

Figure A.9: **Transport experiments on `GAS` benchmark using autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

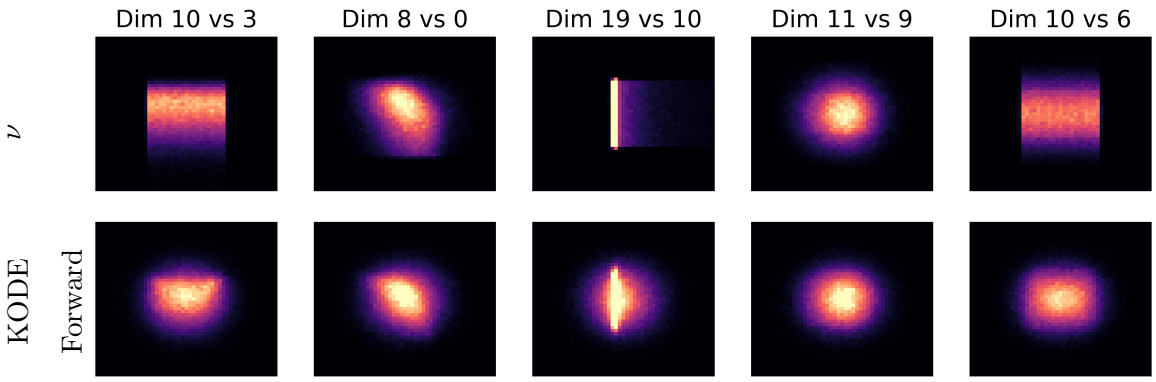

Figure A.10: **Transport experiments on `HEPMASS` benchmark using autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

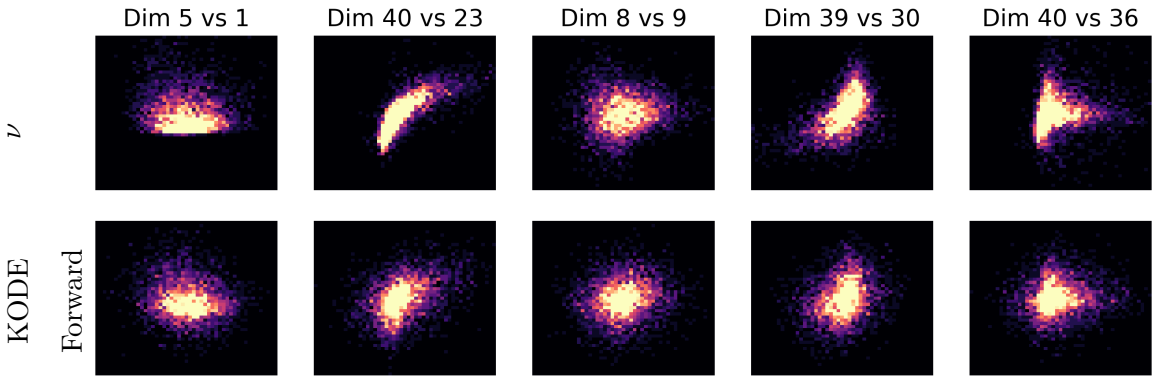

Figure A.11: **Transport experiments on `MINIBOONE` benchmark using autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

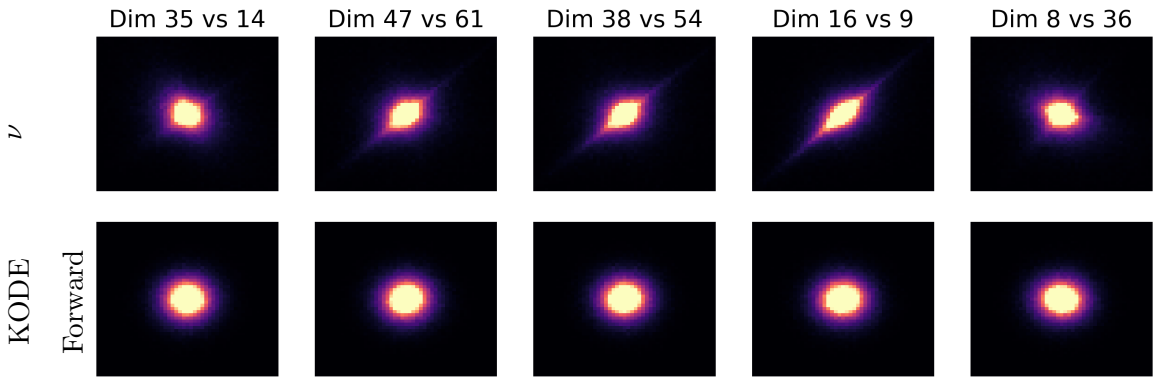

Figure A.12: **Transport experiments on `BSDS300` benchmark using autonomous kernel ODE.** (First row) Marginal distributions of complex data measure ν. (Second row) Marginal distributions of the learned pushforward measure.

Appendix B

STRUCTURED RANDOM RECEPTIVE FIELDS ENABLE INFORMATIVE SENSORY ENCODINGS

B.1 List of abbreviations

Abbreviation	Meaning
ANN	artificial neural network
DCT	discrete cosine transform
DFT	discrete Fourier transform
DHT	discrete Hartley transform
GP	Gaussian process
LN	linear-nonlinear model of a neuron
ReLU	rectified linear unit, $\max(0, x)$ nonlinearity
RFN	random feature network
RKHS	reproducing kernel Hilbert space
SNR	signal-to-noise ratio
STA	spike triggered average
V1	primary visual cortex
XOR	exclusive-or, boolean function

Table B.1: **List of abbreviations**

B.2 Function spaces for wide networks with structured receptive fields

RFNs are intimately connected to a popular class of supervised learning algorithms called kernel methods. As the network width grows, the inner product between the feature representations of

Symbol	Meaning
$\mathbf{x}$	an input or stimulus to the network
$\mathbf{w}$	input-hidden weights for a neuron
β, β_0	readout weights and offset
$y, \hat{y}$	true and predicted output of the network
d	dimension of the stimulus as a vector
m	number of neurons in the hidden layer
T	structured input space
D	dimensions of input space T
$L^2(T)$	space of square-integrable functions over a domain T
ℓ^2	vector space with norm $\|\mathbf{u}\| = \sqrt{\mathbf{u}^T\mathbf{u}}$
$\|\cdot\|$	the L_2 or ℓ_2 norm for function or vector argument
$\|\cdot\|_F$	the Frobenius norm of a matrix
$\langle a, b\rangle$	the $L^2(T)$ inner product between functions, $\langle a, b\rangle = \int_{t\in T} a(t)b(t)\mathrm{d}t$
$\mathbf{u}^T\mathbf{v}$	finite-dimensional ℓ^2 inner product, $\mathbf{u}^T\mathbf{v} = \sum_{i=1}^d u_i v_i$
$\mathbf{I}_d$	$d \times d$ identity matrix
$\mathbf{C}$	$d \times d$ covariance matrix
$\mathcal{H}$	RKHS, comes with inner product $\langle a, b\rangle_{\mathcal{H}}$ and norm $\|a\|_{\mathcal{H}} = \sqrt{\langle a, a\rangle_{\mathcal{H}}}$

Table B.2: **List of important symbols**

two inputs $\mathbf{x}, \mathbf{x}'$ converges to a reproducing kernel

$$k(\mathbf{x}, \mathbf{x}') := \mathbb{E}_{\mathbf{w}}\left[h(\mathbf{x})h(\mathbf{x}')\right]. \tag{B.1}$$

The kernel defines a reproducing kernel Hilbert space (RKHS) of functions. The explicit form of the kernels corresponding to classical RFNs are known for several non-linear activation functions. For example, with the ReLU nonlinearity, no threshold, and unstructured Gaussian weights $\mathbf{w} \sim \mathcal{N}(0, \mathbf{I}_d)$, $k_{\mathrm{ReLU}}(\mathbf{x}, \mathbf{x}') = \frac{1}{\pi}\|\mathbf{x}\|\|\mathbf{x}'\|(\sin\theta + (\pi - \theta)\cos\theta)$ where $\theta = \arccos\left(\frac{\mathbf{x}^T\mathbf{x}'}{\|\mathbf{x}\|\|\mathbf{x}'\|}\right)$ [50].

We derive the kernel induced by our RFNs with hidden weights initialized from GPs. In this

section we work in the discrete setting, but the continuous version is analogous. Recall the network equations that we use

$$\mathbf{h} = \sigma\left(\mathbf{W}\mathbf{x}\right), \qquad \hat{y} = \beta^T \mathbf{h} + \beta_0, \tag{B.2}$$

and the basis change Theorem,

Theorem B.1 (Basis change formula)**.** *Assume* $\mathbf{w} \sim \mathcal{N}(0, \mathbf{C})$ *with* $\mathbf{C} = \mathbf{\Phi}\mathbf{\Lambda}^2\mathbf{\Phi}^T$ *its eigenvalue decomposition. For* $\mathbf{x} \in \mathbb{R}^d$*, define*

$$\tilde{\mathbf{x}} := \mathbf{\Lambda}\mathbf{\Phi}^T\mathbf{x}. \tag{B.3}$$

Then $\mathbf{w}^T\mathbf{x} = \mathbf{z}^T\tilde{\mathbf{x}}$ *for* $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}_d)$*.*

By definition of the kernel Eq (B.1), network Eq (B.2), and Theorem B.1, the kernel for structured features

$$\begin{aligned} k_{\text{struct}}(\mathbf{x}, \mathbf{x}') &= \mathbb{E}_{\mathbf{w}\sim\mathcal{N}(0,\mathbf{C})}\left[h(\mathbf{x})h(\mathbf{x}')\right] \\ &= \mathbb{E}_{\mathbf{w}\sim\mathcal{N}(0,\mathbf{C})}\left[\sigma(\mathbf{w}^T\mathbf{x})\sigma(\mathbf{w}^T\mathbf{x}')\right] \\ &= \mathbb{E}_{\mathbf{z}\sim\mathcal{N}(0,\mathbf{I}_d)}\left[\sigma\left(\mathbf{z}^T\tilde{\mathbf{x}}\right)\sigma\left(\mathbf{z}^T\tilde{\mathbf{x}}'\right)\right] \\ &:= k_{\text{unstruct}}(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}'). \end{aligned} \tag{B.4}$$

Thus, the induced kernels from structured weights can be found in terms of unstructured weight kernels acting on the transformed inputs $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$. Taking ReLU as the nonlinearity for example, we get that $k_{\text{struct}}(\mathbf{x}, \mathbf{x}') = k_{\text{ReLU}}(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')$.

Every RKHS $\mathcal{H}$ comes with an inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ and norm $\|\cdot\|_{\mathcal{H}} = \sqrt{\langle\cdot,\cdot\rangle_{\mathcal{H}}}$. The norm and inner product can be expressed in terms of eigenvalues and eigenfunctions of the kernel itself, analogous to the eigendecomposition of the covariance function of the GP weights. Although it is beyond the scope of our paper to explain the theory in detail, there are well-established results showing that functions with small $\mathcal{H}$-norm are easier to learn than those with larger norm for a wide variety of kernel-based algorithms [197, 196]. In ridge regression, this effect is again equivalent to projection and filtering in the kernel eigenbasis, i.e. linear filtering in function space. Finally, end-to-end trained networks where the weights $\mathbf{W}$ are optimized may be studied with the related neural tangent kernel (NTK) when the step size is small [114]. The basis change Theorem B.1 and Eq (B.4) give us a way to understand the RKHS of the stuctured network in terms of an unstructured network's RKHS acting on the transformed inputs $\tilde{\mathbf{x}}$.

Kernel eigenfunctions differ with structured weights

The structured RKHS has eigenfunctions which are different from the eigenfunctions of the unstructured RKHS. To see this, it's necessary to introduce a probability measure $\mu(\mathbf{x})$ for the data $\mathbf{x} \in \mathbb{R}^d$. Kernel learning is often understood [40] in the orthonormal basis for $L^2(\mu)$ given by the eigenfunctions ψ_i of the integral operator $\mathcal{T}_k$ defined by

$$(\mathcal{T}_k f)(\mathbf{x}) = \int k(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') \mathrm{d}\mu(\mathbf{x}').$$

A natural question to ask is, how does the eigensystems of $\mathcal{T}_{\text{struct}}$ and $\mathcal{T}_{\text{unstruct}}$ compare? The mapping from $\mathbf{x}$ to $\tilde{\mathbf{x}}$ induces a pushforward measure on $\tilde{\mathbf{x}}$ which we will call $\nu(\tilde{\mathbf{x}})$, and since the mapping is linear $\nu(\tilde{\mathbf{x}}) = \mu(\mathbf{\Phi}^T \mathbf{\Lambda}^{-1} \mathbf{x})$. (Note that $\mathrm{d}\nu(\mathbf{x}) = |\mathbf{\Lambda}|^{-1} \mathrm{d}\mu(\mathbf{\Phi}^T \mathbf{\Lambda}^{-1} \mathbf{x})$, and if μ is multivariate Gaussian, then ν is Gaussian with a different covariance.) Thus the integral operator $\mathcal{T}_{\text{struct}}$ under the data measure is equivalent to integrating kernel k_{unstruct} under the pushforward measure. Because

$$\int k_{\text{unstruct}}(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') \mathrm{d}\mu(\mathbf{x}') \qquad \text{and} \qquad \int k_{\text{unstruct}}(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') \mathrm{d}\nu(\mathbf{x}')$$

are different, there is no general relationship that holds between the eigenfunctions. Different measures leading to different kernel eigenfunctions and eigenvalues can explain why structured weights have strong effects on learning, as shown by recent work demonstrating that areas of low input density are learned more slowly [18].

B.3 Kernel theory for frequency detection: a fully worked, highly nutritious, simplified example

To demonstrate how the kernel theory can also explain the benefits of structured random features, we developed a simplified frequency detection task. We fully explain the orthonormal basis, target function in that basis, kernels, and how the target function is harder to learn without knowledge of the structure. To our knowledge, this is the first technical result that explains how transforming the input data into an informative, lower-dimensional representation (essentially, preprocessing) leads to improved learning with kernel methods.

Data distribution and orthonormal basis

Frequency detection as presented in Appendix B.8 is difficult to work with because of the different distributions of x_+ and x_- samples. Kernel theory requires us to work with basis functions which are orthonormal with respect to the data measure, which in that case would be a mixture. To get around this issue, we simplify the data distribution so that all of our data are Gaussian white noise $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I}_d)$. The Gaussian measure in 1-d has the natural orthogonal basis of (probabilist's) Hermite polynomials $He_k(x)$ [1]. The first few of these are

$$He_0(x) = 1, \quad He_1(x) = x, \quad He_2(x) = x^2 - 1.$$

We can normalize these as $\psi_k(x) = (k!)^{-1/2} He_k(x)$ so that orthonormality under the Gaussian distribution

$$\int_{-\infty}^{\infty} \psi_k(x)\psi_{k'}(x) \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \mathrm{d}x = \delta_{k,k'} \tag{B.5}$$

is satisfied. To construct an orthonormal basis in $d > 1$ dimensions, we take the tensor product of Hermite polynomials,

$$\psi_{\mathbf{k}}(\mathbf{x}) = \prod_{i=1}^{d} \psi_{k_i}(x_i), \tag{B.6}$$

which is orthonormal for the multivariate Gaussian

$$\int_{\mathbb{R}^d} \psi_{\mathbf{k}}(\mathbf{x})\psi_{\mathbf{k}'}(\mathbf{x})(2\pi)^{-d/2} e^{-\|\mathbf{x}\|^2/2} \mathrm{d}\mathbf{x} = \delta_{\mathbf{k},\mathbf{k}'} \tag{B.7}$$

due to the separability of the integrals over each dimension.

Target function

The labels $y = f^*(\mathbf{x})$ are chosen to only depend on the amplitude of frequency f_1 in the white noise signal. Writing $\tilde{\mathbf{x}} = \mathbf{\Phi}^T \mathbf{x}$ for the input signal in frequency space, with $\mathbf{\Phi}$ the discrete cosine transform (DCT) matrix [206], the target function f^* only depends on $|\tilde{x}_{f_1}|$. A simplified classfication task would then use

$$f^*(\mathbf{x}) = \begin{cases} +1 & \text{if } |\tilde{x}_{f_1}| \geq \theta \\ -1 & \text{if } |\tilde{x}_{f_1}| < \theta \end{cases}, \tag{B.8}$$

where the threshold θ is chosen so that 50% of the points are labeled ± 1. However, we need to write the target function f^* in the polynomial basis Eq (B.6), and Eq (B.8) is a step function which has an infinite polynomial expansion. A much simpler target function

$$f^*(\mathbf{x}) = |\tilde{x}_{f_1}|^2 - \theta^2 \tag{B.9}$$

still captures some of the behavior of Eq (B.8): It's negative when the power in frequency f_1 is below θ and positive above. We will analyze the quadratic surrogate Eq (B.9).

Kernel decomposition

We consider kernels $k_{\text{struct}}, k_{\text{unstruct}}$ of the form

$$\kappa(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{k}} \rho_{\mathbf{k}} \psi_{\mathbf{k}}(\mathbf{x}) \psi_{\mathbf{k}}(\mathbf{x}'), \tag{B.10}$$

where $\mathbf{k} = (k_1, k_2, \ldots, k_d)$ is a multi-index. We build κ out of 1-d kernels

$$\kappa_i(x_i, x_i') = \sum_{k_i=0}^{\infty} \rho_{k_i} \psi_{k_i}(x_i) \psi_{k_i}(x_i'), \tag{B.11}$$

where the eigenvalue sequence $\rho_0, \rho_1, \ldots$ is shared across different κ_i. Each κ_i defines an RKHS over functions of a single variable $\mathcal{H}_i$, and $\kappa(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^d \kappa_i(x_i, x_i')$ defines an RKHS which is the tensor product $\mathcal{H} = \otimes_{i=1}^d \mathcal{H}_i$ with eigenvalues given as $\rho_{\mathbf{k}} = \prod_{i=1}^d \rho_{k_i}$. These tensor product polynomial kernels were considered in [12] for multiple kernel learning. We normalize the 1-d kernels Eq (B.11) so that $\sum_{k=0}^{\infty} \rho_k = 1$. This implies that

$$\sum_{\mathbf{k}} \rho_{\mathbf{k}} = \sum_{k_1, \ldots, k_d = 0}^{\infty} \prod_{i=1}^{d} \rho_{k_i} = \left(\sum_{k_1=0}^{\infty} \rho_{k_1} \right) \cdots \left(\sum_{k_d=0}^{\infty} \rho_{k_d} \right) = 1, \tag{B.12}$$

by the separability of the eigenvalues, so that the d-dimensional kernel Eq (B.10) is also normalized. Furthermore, we can assume that all of the $\rho_i < 1$.

In general, random feature kernels will not be tensor products unless we make very specific choices of weights and nonlinearities. We only use the product structure for convenience with normalization. The form Eq (B.10) is, on the other hand, quite general and will hold for a variety of random feature kernels.

Target function in unstructured and structured networks

Now we analyze the target function in both unstructured and unstructured spaces. Using structured random feature map is equivalent to some deterministic remapping $\tilde{\mathbf{x}} = \mathbf{\Lambda}\mathbf{\Phi}^T\mathbf{x}$, where $\mathbf{\Phi}$ is an orthogonal matrix encompassing the basis change and $\mathbf{\Lambda}$ is a filtering matrix. Like in Section B.9, let's use stationary bandpass features so that $\mathbf{\Lambda}$ just contains d' entries which are equal to 1 with the rest 0, and $\mathbf{\Phi}$ is the DCT. Since the DCT is unitary, $\tilde{\mathbf{x}} \sim \mathcal{N}(0, I_{d'})$. We use the DCT rather than the DFT to avoid complications dealing with complex variables. Thus the transformed variables $\tilde{\mathbf{x}}$ and the original variables $\mathbf{x}$ both follow similar spherical Gaussian distributions just in different dimensions. This means that the Hermite polynomial basis Eq (B.6) is an orthonormal basis for both spaces.

Let $\tilde{\mathcal{H}}$ be the RKHS of functions after applying the transformation $\mathbf{x} \mapsto \tilde{\mathbf{x}}$. Then $\tilde{\mathcal{H}}$ has the kernel $\kappa(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')$ and eigenbasis $\psi_{\mathbf{k}}(\tilde{\mathbf{x}})$ under measure $\tilde{\mathbf{x}} \sim \mathcal{N}(0, I_{d'})$. We take $\mathcal{H}$ to be the RKHS of functions without ever transforming coordinates, i.e. the RKHS with kernel $\kappa(\mathbf{x}, \mathbf{x}')$ with eigenbasis $\psi_{\mathbf{k}}(\mathbf{x})$ under data measure $\mathbf{x} \sim \mathcal{N}(0, I_d)$.

The target function Eq (B.9) is simple to express in the eigenbasis of $\tilde{\mathcal{H}}$:

$$\begin{aligned} f^*(\tilde{\mathbf{x}}) &= |\tilde{x}_{f_1}|^2 - \theta^2 \\ &= He_2(\tilde{x}_{f_1}) - (\theta^2 - 1)He_0(\tilde{x}_{f_1}) \\ &= \sqrt{2}\psi_2(\tilde{x}_{f_1}) - (\theta^2 - 1)\psi_0(\tilde{x}_{f_1}). \end{aligned}$$

Any RKHS $\mathcal{F}$ with eigenvalues ρ_i and eigenfunctions ψ_i defines an inner-product

$$\langle f, g\rangle_{\mathcal{F}} = \sum_i \frac{\langle f, \psi_i\rangle_{L^2(\mu)}\langle g, \psi_i\rangle_{L^2}(\mu)}{\rho_i}, \tag{B.13}$$

where the $L^2(\mu)$ inner product is with respect to the data measure μ and $\|f\|_{\mathcal{F}}^2 = \langle f, f\rangle_{\mathcal{F}}$ [11]. The norm of our target function thus becomes

$$\|f^*\|_{\tilde{\mathcal{H}}}^2 = \frac{2}{\rho_2\rho_0^{d'-1}} + \frac{(\theta^2 - 1)^2}{\rho_0^{d'}}. \tag{B.14}$$

We will see that working with the less informative RKHS $\mathcal{H}$ leads to a significantly larger norm.

To express the target function Eq (B.9) in the eigenbasis of $\mathcal{H}$ we must use the basis change

matrix. Let $\mathbf{u}$ be the vector corresponding to row f_1 of $\mathbf{\Phi}^T$ so that $\tilde{x}_{f_1} = \mathbf{u}^T\mathbf{x} = \sum_{i=1}^d u_i x_i$. Thus,

$$\begin{aligned} f^*(\mathbf{x}) &= |\tilde{x}_{f_1}|^2 - 1 \\ &= \left(\sum_{i=1}^d u_i x_i\right)^2 - \theta^2 \\ &= \sum_{i=1}^d u_i^2 x_i^2 + 2\sum_{i<j} u_i u_j x_i x_j - \theta^2 \\ &= \sum_{i=1}^d u_i^2(\sqrt{2}\psi_2(x_i) + 1) + 2\sum_{i<j} u_i u_j \psi_1(x_i)\psi_1(x_j) - \theta^2. \end{aligned}$$

Reading off the coefficients of the basis terms (recall that $1 = \psi_{\mathbf{0}}(\mathbf{x})$), we get that

$$\begin{aligned} \|f^*\|_{\mathcal{H}}^2 &= \frac{2}{\rho_2\rho_0^{d-1}}\left(\sum_{i=1}^d u_i^4\right) + \frac{1}{\rho_0^d}\left(\sum_{i=1}^d u_i^2 - \theta^2\right)^2 + \frac{4}{\rho_1^2\rho_0^{d-2}}\sum_{i<j}(u_i u_j)^2 \\ &= \frac{2}{\rho_2\rho_0^{d-1}}\left(\sum_{i=1}^d u_i^4\right) + \frac{(\theta^2-1)^2}{\rho_0^d} + \frac{4}{\rho_1^2\rho_0^{d-2}}\sum_{i<j}(u_i u_j)^2, \end{aligned} \tag{B.15}$$

where the $\sum_{i=1}^d u_i^2$ terms are equal to 1 because $\mathbf{\Phi}$ is unitary. Note that

$$1 = \|\mathbf{u}\mathbf{u}^T\|_F^2 = \sum_{i,j}(u_i u_j)^2 = 2\sum_{i<j}(u_i u_j)^2 + \sum_{i=1}^d u_i^4, \tag{B.16}$$

so the terms involving sums combined are $O(\rho_0^d)$. Specifically for the DCT-II [206], components $u_i = \sqrt{\frac{2}{d}}\cos\left(\frac{\pi f_1}{d}\left(i - \frac{1}{2}\right)\right)$, so for any $f_1 \in \{1, \ldots (d-1)\}$

$$\sum_{i=1}^d u_i^4 = \frac{4}{d^2}\sum_{k=0}^{d-1}\cos^4\left(\frac{\pi f_1}{d}\left(k + \frac{1}{2}\right)\right) = \frac{3}{2d},$$

which implies that $\sum_{i<j}(u_i u_j)^2 = \frac{1}{2} - \frac{3}{4d}$. (For general unit vectors $\mathbf{u}$ we have that $1 \geq \sum_{i=1}^d u_i^4 \geq d^{-1}$ by Hölder's inequality.) Thus for DCT-II the norm is

$$\|f^*\|_{\mathcal{H}}^2 = \frac{3}{d\rho_2\rho_0^{d-1}} + \frac{(\theta^2-1)^2}{\rho_0^d} + \frac{2}{\rho_1^2\rho_0^{d-2}}\left(1 - \frac{3}{2d}\right). \tag{B.17}$$

Comparison of the learning performance in structured and unstructured RKHS

A very standard but rough bound on the generalization performance of kernel ridge regression or classification can be found by analyzing the Rademacher complexity of the class of linear functions

in the RKHS [196, 197]. These upper bound the expected loss over new data in terms of the training loss plus an error term, c.f. Theorem 7.39 in [197] for kernel ridge regression. The error term controls the generalization gap between test and training losses and is typically proportional to the $\|f^*\|_{\mathcal{F}}$ (for 0/1 loss) or $\|f^*\|_{\mathcal{F}}^2$ (for square loss).[1] Thus, a function with small $\mathcal{F}$-norm is *easier to learn* in the *precise sense* that it takes a smaller training set size to achieve a given generalization gap when the norm is smaller.

We have computed $\|f^*\|_{\mathcal{H}}^2$ and $\|f^*\|_{\tilde{\mathcal{H}}}^2$ in Eq (B.15) and Eq (B.14). Examining those two expressions, we see that $\|f^*\|_{\mathcal{H}}^2 \gg \|f^*\|_{\tilde{\mathcal{H}}}^2$ due to two factors: First, there is the splitting of the quadratic target into x_i^2 and $x_i x_j$ terms which creates the extra term $\frac{4}{\rho_1^2 \rho_0^{d-2}} \sum_{i<j} (u_i u_j)^2$ in the norm. This arises from *nonlinear effects* of the quadratic target function on the different components of the input signal. This nonlinear term is $O(\rho_0^{-d})$, the same order as the other terms, but Eq (B.15) will be larger than Eq (B.14) for $d' = d$ when $\rho_1^2 < \rho_0 \rho_2$. Secondly, the *dimension reduction* from d to d' dimensions means that analogous terms in Eq (B.15) and Eq (B.14) scale like c^d and $c^{d'}$, respectively. This means that $\|f^*\|_{\mathcal{H}}^2 \geq c^{(d-d')} \|f^*\|_{\tilde{\mathcal{H}}}^2$. Note that the projection $\mathbf{x} \mapsto \tilde{\mathbf{x}}$ is a linear operation but has nonlinear consequences for the norm. The norm in the unstructured kernel space is exponentially larger than the norm in the structured kernel space.

Interpreting this exponential norm separation in light of the Rademacher generalization bounds, you would need to train on exponentially more samples with the unstructured versus the structured kernel to acheive the same bound. The majority of this effect is due to the dimension reduction factor, since the nonlinear factor only grows the norm by a constant. However, for target functions with contributions from many higher-order polynomials, the nonlinear factor would have a non-linearly stronger effect. This is an interesting avenue for future research. More precise estimates of generalization performance are possible using theories such as [35]. However, the exponential gap between the performance of these two kernels is fundamental and not an artificial result from shortcomings of the Rademacher analysis.

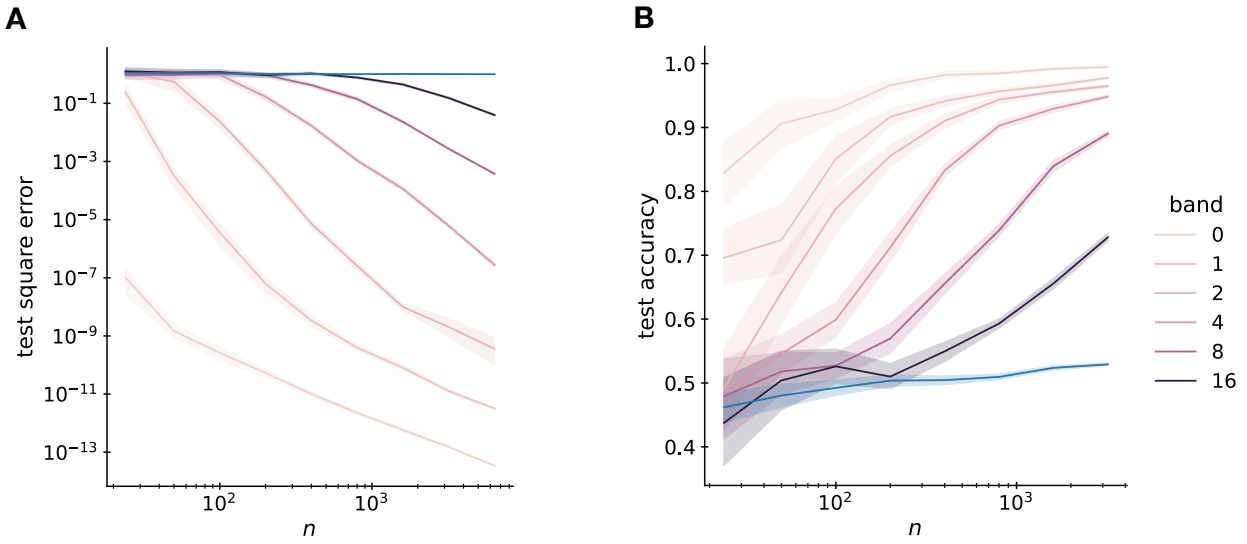

Figure B.1: **Simulation results for the simplified frequency detection task.** On the left, test error versus dataset size for kernel ridge regression using structured kernels (purple lines, by bandwidth) and the unstructured kernel (blue). On the right, test accuracy versus dataset size for SVM classifier readout trained on structured random features (purple lines, by bandwidth) and unstructured random features (blue). In both cases, task structural information improves performance, leading to less error and higher accuracy.

Simulation results support the theory

We ran simulations of the simplified frequency detection task with both kernel methods and random feature networks to check that this simplified task was similar to the task presented in the main text and Appendix B.8. These simulations were performed with regression tasks with targets like Eq (B.9) and classification tasks equivalent to Eq (B.8). The results are consistent for a broad range of estimators and for both kernel and random feature networks.

For regression tasks, we used a tensor product kernel of the form Eq (B.10) with kernel ridge

[1]The trace of the kernel matrix also appears in these bounds, but this is unity due to normalization Eq (B.12).

regression and the target function $f^*(\mathbf{x}) = a(|\tilde{x}_{f_1}|^2 - \theta^2)$. The constant a was chosen so that the labels had standard deviation 1, meaning that a mean square error of 1 is equivalent to chance. For varying n logarithmically spaced between 24 and 6,400 we generated training and testing sets of size $n/2$ in d =1,000 dimensions with $f_1 = 16$. The training set was used to select the ridge parameter from powers of 10 in the range $[10^{-3}, \ldots, 10^3]$ by 5-fold cross-validation and error was computed on the test set. Each of these experiments was repeated 20 times. We compare unstructured and structured kernels by either passing in the raw vectors $\mathbf{x}$ (unstructured) or first performing the DFT and passing in $(\tilde{\mathbf{x}})_{f_1 - \texttt{band}:f_1 + \texttt{band}}$. Here, `band` is a bandwidth parameter determining the number of components that are used in the structured kernel. When `band` = 0 we keep only the target frequency component.

The classification tasks were similar. We used the exact target function Eq (B.8), the range of n from 24 to 3,200, same split into training and test sets, $d = 100$, and $f_1 = 16$. We classified using a random feature network with 2,000 neurons, ReLU nonlinearity, and zero bias trained with a linear SVM classifier readout. The SVM regularization strength was also swept over the range $[10^{-3}, \ldots, 10^3]$ by powers of 10 and selected via 5-fold cross-validation on the training set. The accuracy of the selected estimator was then computed on the test set. We compare both classical and mechanosensory receptive fields with bandwidth parameter `band` equivalent to $f_{\text{lo}} = f_1 - \texttt{band}$ and $f_{\text{hi}} = f_1 + \texttt{band}$ and no decay ($\gamma = \infty$).

The results are shown in Fig. B.1. For either regression or classification, the best performance is acheived by the structured method with smallest bandwidth: error increases and accuracy degrades with bandwidth. The unstructured kernel and unstructured random feature methods both only perform at chance levels, square error ≈ 1 or accuracy ≈ 0.5. Similar results occur with kernel SVM on the classification task, kernel support vector regression, and for kernels which are not tensor product kernels (not shown). Furthermore, the random feature network's limiting kernel is an arc-cosine kernel which is not a tensor product kernel.

These simulations show that simplified frequency detection still exhibits the main features of the other tasks we study while remaining amenable to theoretical analysis. The simulation results are not sensitive to details of the experiment and are qualitatively similar when assumptions of the theory are broken. Getting quantitative predictions of training and testing errors from the theory

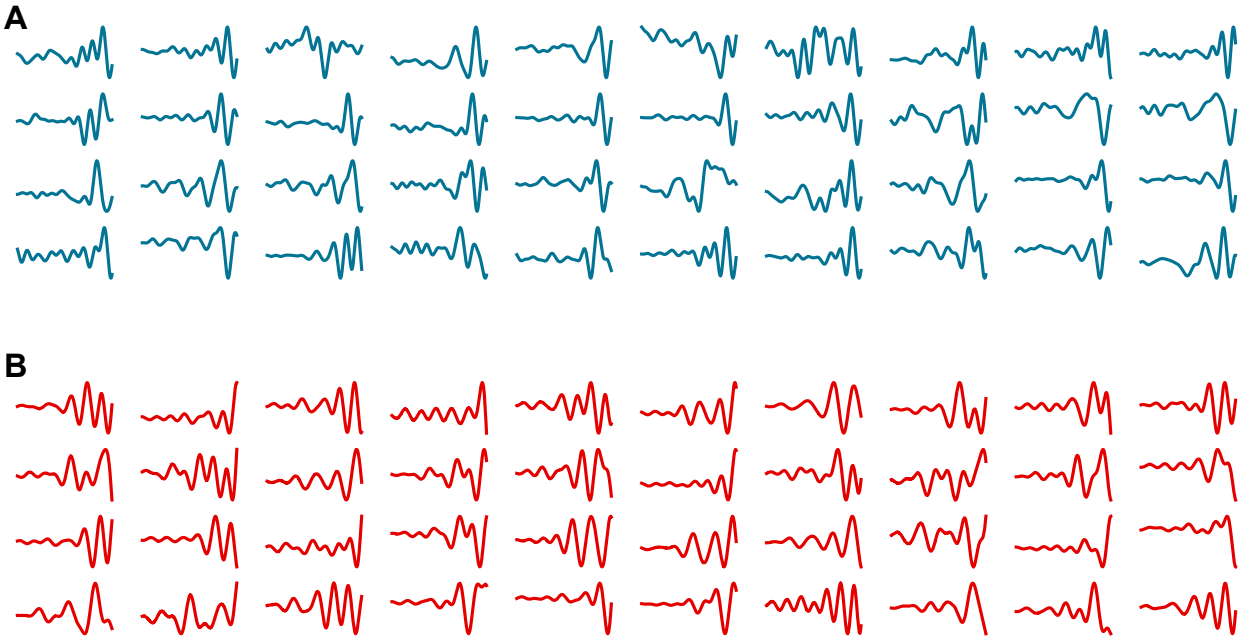

Figure B.2: **Receptive fields of mechanosensory neurons.** We show (A) biological receptive fields and (B) random samples from the fitted covariance model.

is left to future work.

B.4 Covariance parameter optimization

Here we describe the details of how the GP covariances were fit to our various datasets.

Mechanosensor covariance

We aim to minimize the difference between the matrix generated by the covariance model $\mathbf{C}_{\text{model}}$ and the data $\mathbf{C}_{\text{data}}$, while keeping f_{lo} smaller than f_{hi}. For simplicity, we measure the covariance mismatch with the Frobenius norm, solving

$$\min_{f_{\text{lo}}, f_{\text{hi}}, \gamma} \|\mathbf{C}_{\text{model}}(f_{\text{lo}}, f_{\text{hi}}, \gamma) - \mathbf{C}_{\text{data}}\|_F \tag{B.18}$$
$$\text{subject to: } f_{\text{hi}} \geq f_{\text{lo}}.$$

We use the trust region algorithm provided by the `scipy.optimize.minimize` to solve Eq (B.18).

V1 covariance

To fit the covariance model to the data, we formulate an optimization problem over the model parameters s and f, where we minimize the Frobenius norm of the difference between the covariance matrix $\mathbf{C}_{\text{model}}$ and $\mathbf{C}_{\text{data}}$:

$$\min_{s,f} \|\mathbf{C}_{\text{model}}(s,f) - \mathbf{C}_{\text{data}}\|_F. \tag{B.19}$$

We solve Eq (B.19) using the Broyden–Fletcher–Goldfarb–Shannon (BFGS) algorithm provided by the

`scipy.optimize.minimize` package.

B.5 Null receptive field models

We construct null GP models for both mechanosensors and V1 for comparison.

Mechanosensor covariance

We compare the data covariance matrix with the unstructured model and the Fourier model:

$$C(t,t') = \overbrace{\sum_{k=0}^{\infty} \lambda_k^2 \cos\left(\omega_k(t-t')\right)}^{\text{stationary process}}, \qquad \lambda_k = \overbrace{\begin{cases} 1 & f_{\text{lo}} \le \omega_k \le f_{\text{hi}} \\ 0 & \text{otherwise} \end{cases}}^{\text{bandlimited, flat-power spectrum}} \tag{B.20}$$

For the Fourier model, we fit the f_{hi} and f_{lo} parameters to data by minimizing the Frobenius error in the covariance matrix, finding $f_{lo} = 75$ Hz, $f_{hi} = 200$ Hz. The resulting covariance matrices are shown in Fig. B.3 and Fig. B.4. Receptive field samples from the null models are shown in Fig. B.5. The samples from the Fourier model are smooth but do not decay in time like biological receptive fields.

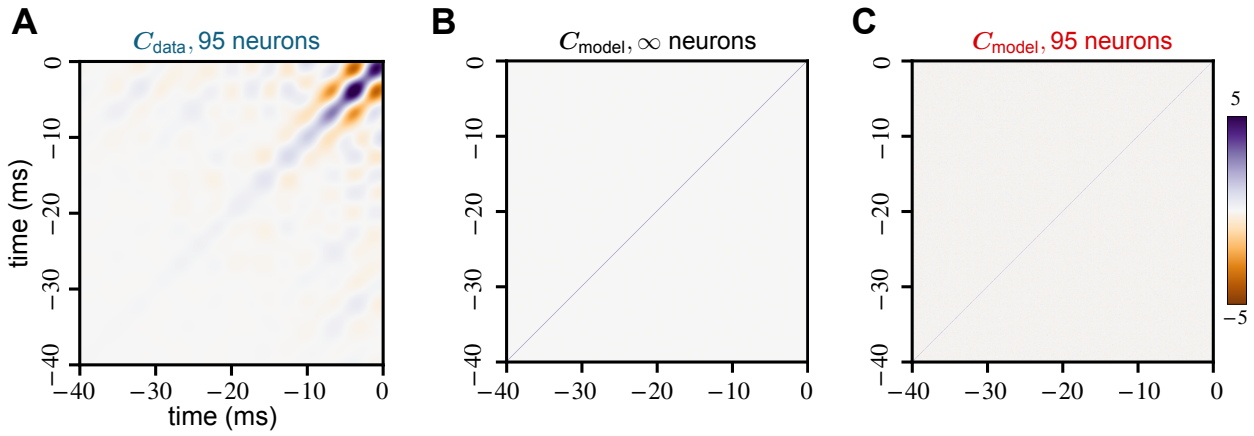

Figure B.3: **Covariance matrix of mechanosensory receptive fields and unstructured model.** We compare the covariance matrices generated from the (A) receptive fields of 95 mechanosensory neurons, (B) unstructured GP model and (C) 95 random samples from the model.

V1 covariance

We compare the data covariance matrix with the unstructured model and the translation invariant version of the V1 model. Recall that the localized V1 model had covariance

$$C(\mathbf{t},\mathbf{t}') = \overbrace{\exp\left(-\frac{\|\mathbf{t}-\mathbf{t}'\|^2}{2f^2}\right)}^{\text{smooth receptive fields}} \cdot \overbrace{\exp\left(-\frac{\|\mathbf{t}-\mathbf{c}\|^2+\|\mathbf{t}'-\mathbf{c}\|^2}{2s^2}\right)}^{\text{localized to a center } c}. \tag{B.21}$$

For the translation invariant model, we remove the localizing exponential and only fit the spatial frequency parameter, f (finding $f = 0.73$ pixels). The neuron weights are generated from a covariance function of the following form:

$$C(\mathbf{t},\mathbf{t}') = \overbrace{\exp\left(-\frac{\|\mathbf{t}-\mathbf{t}'\|^2}{2f^2}\right)}^{\text{smooth receptive fields}} \tag{B.22}$$

The resulting covariance matrices are shown in Fig. B.8 and Fig. B.9. Receptive field samples from the null models are show in Fig. B.10.

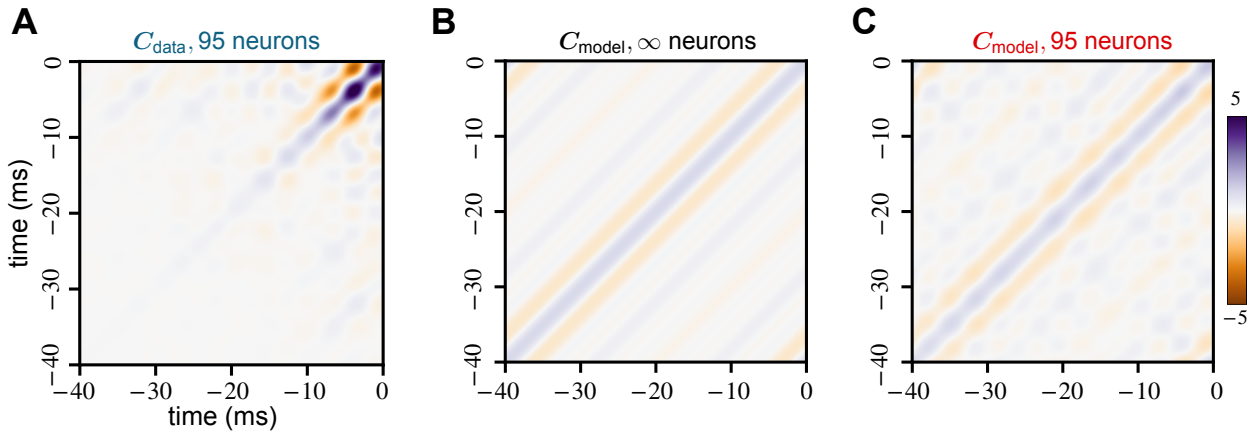

Figure B.4: **Covariance matrix of mechanosensory receptive fields and the Fourier model (B.20).** We compare the covariance matrices generated from the (A) receptive fields of 95 mechanosensory neurons, (B) Fourier GP model and (C) 95 random samples from the model.

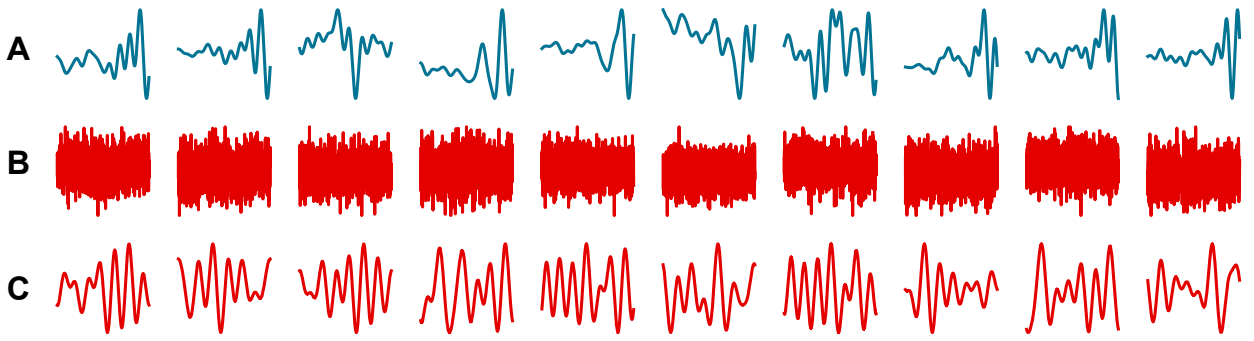

Figure B.5: **Receptive fields from mechanosensory neurons, the unstructured model and the Fourier model (B.20).** We show the receptive fields from the (A) mechanosensory neurons, (B) unstructured GP model and (C) the Fourier GP model.

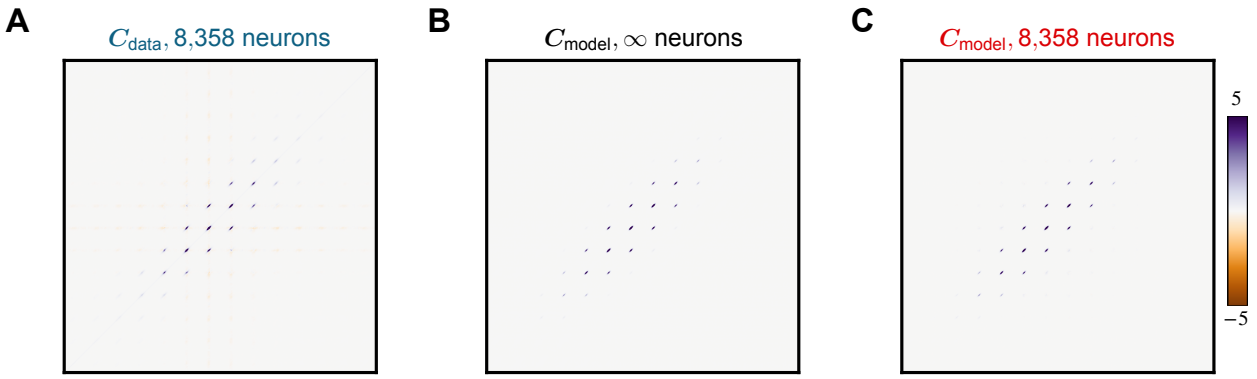

Figure B.6: **Covariance matrix of V1 receptive fields and our model for white noise stimuli.** We show the full structure of the covariance matrices, which are the 180×180 pixel region around the centers of these 504×504 pixel matrices. .These matrices are generated from the (A) receptive fields of 8,358 mouse V1 neurons, (B) the GP model Eq (B.21), and (C) 8,358 random samples from the model.

B.6 Derivation of eigenfunctions of V1 covariance function

The covariance between two pixel locations $\mathbf{t} = (t_1, t_2), \mathbf{t}' = (t_1', t_2') \in \mathbf{R}^2$ is given by

$$\begin{aligned} C(\mathbf{t}, \mathbf{t}') &= e^{-\frac{\|\mathbf{t}-\mathbf{t}'\|^2}{2f^2}} e^{-\frac{\|\mathbf{t}\|^2+\|\mathbf{t}'\|^2}{2s^2}} \\ &= e^{-\frac{(t_1-t_1')^2}{2f^2}} e^{-\frac{(t_1^2+t_1'^2)}{2s^2}} \cdot e^{-\frac{(t_2-t_2')^2}{2f^2}} e^{-\frac{(t_2^2+t_2'^2)}{2s^2}} \\ &= \prod_{i=1}^{2} e^{-\alpha(t_i-t_i')^2} e^{-\delta(t_i^2+t_i'^2)} \qquad \text{for } \alpha := \frac{1}{2f^2},\ \delta := \frac{1}{2s^2}. \end{aligned}$$

Since covariance function factors into a product of functions of variables t_1, t_1' and t_2, t_2', the multidimensional eigenfunctions $\phi_{\mathbf{k}}(\mathbf{t})$ and eigenvalues $\lambda_{\mathbf{k}}^2$ also factor into a product of 1-dimensional eigenfunction and eigenvalues, i.e. $\phi_{\mathbf{k}}(\mathbf{t}) = \prod_{i=1}^{2} \phi_{k_i}(t_i)$ and $\lambda_{\mathbf{k}}^2 = \prod_{i=1} \lambda_{k_i}^2$. This holds for $d > 2$ dimensions as well. So we work in 1-d and search for eigenfunctions and eigenvalues such that,

$$\int_{-\infty}^{\infty} C(t, t')\phi_k(t)dt = \lambda_k^2 \phi_k(t'),$$

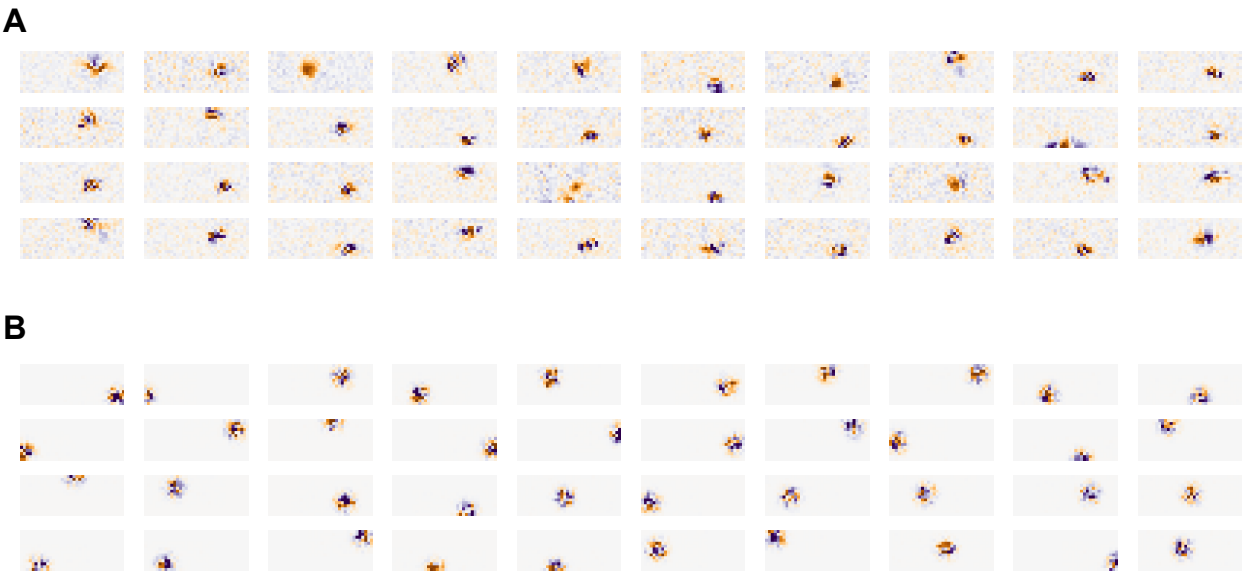

Figure B.7: **Receptive fields of V1 neurons from white noise stimuli.** We show (A) biological receptive fields and (B) random samples from the fitted covariance model.

with $C(t,t') = e^{-\alpha(t-t')^2} e^{-\delta(t^2+t'^2)}$.

We make the ansatz that $\phi_k(t) = e^{-c_1 t^2} H_k(c_2 t)$, where H_k is the k^{th} Hermite polynomial (physicists' convention) [92] and c_1, c_2 are constants. With this guess for the eigenfunctions,

$$\begin{aligned}
\int_{-\infty}^{\infty} C(t,t')\phi_k(t)dt &= \int_{-\infty}^{\infty} e^{-\alpha(t-t')^2} e^{-\delta(t^2+t'^2)} e^{-c_1 t^2} H_k(c_2 t)dt \\
&= \int_{-\infty}^{\infty} e^{-Xt^2+Yt-t'^2(\alpha+\delta)} H_k(c_2 t)dt && (X := \alpha+\delta+c_1, Y := 2\alpha t') \\
&= e^{-t'^2(\alpha+\delta)+\frac{Y^2}{4X}} \int_{-\infty}^{\infty} e^{-(\sqrt{X}t-\frac{Y}{2\sqrt{X}})^2} H_k(c_2 t)dt && \text{(completing the square)} \\
&= \frac{1}{\sqrt{X}} e^{-t'^2(\alpha+\delta)+\frac{Y^2}{4X}} \int_{-\infty}^{\infty} e^{-\left(u-\frac{Y}{2\sqrt{X}}\right)^2} H_k(u\frac{c_2}{\sqrt{X}}) && (u = \sqrt{X}t) \\
&= \underbrace{\sqrt{\frac{\pi}{X}}\left(1-\frac{c_2^2}{X}\right)^{k/2}}_{\lambda_k^2} \underbrace{e^{-t'^2(\alpha+\delta-\frac{\alpha^2}{X})} H_k\left(\frac{c_2\alpha}{X(1-\frac{c_2^2}{X})^{1/2}}t'\right)}_{\hat{\phi}_k(t')}. && \text{([92], 7.374.8)}
\end{aligned}$$

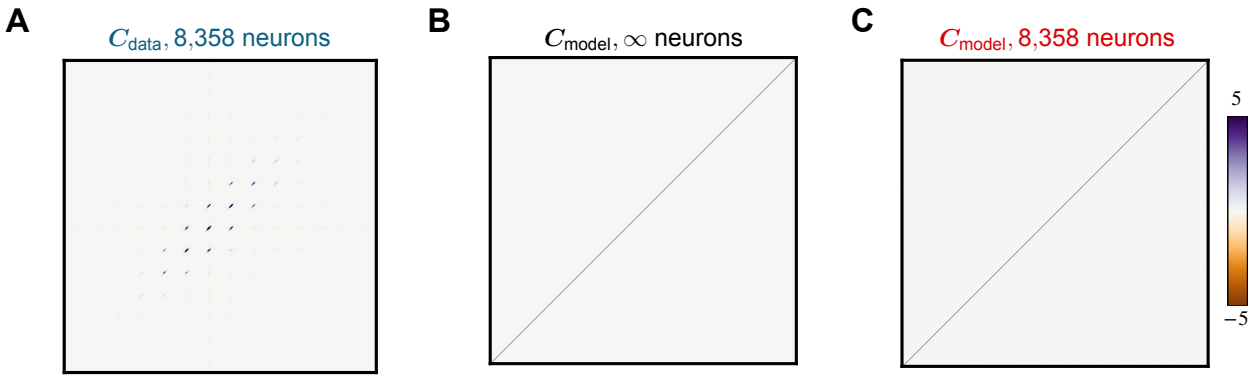

Figure B.8: **Covariance matrix of V1 receptive fields and unstructured model for white noise stimuli.** We compare the covariance matrices generated from the (A) receptive fields of 8,358 mice V1 neurons, (B) unstructured GP model and (C) 8,358 random samples from the model.

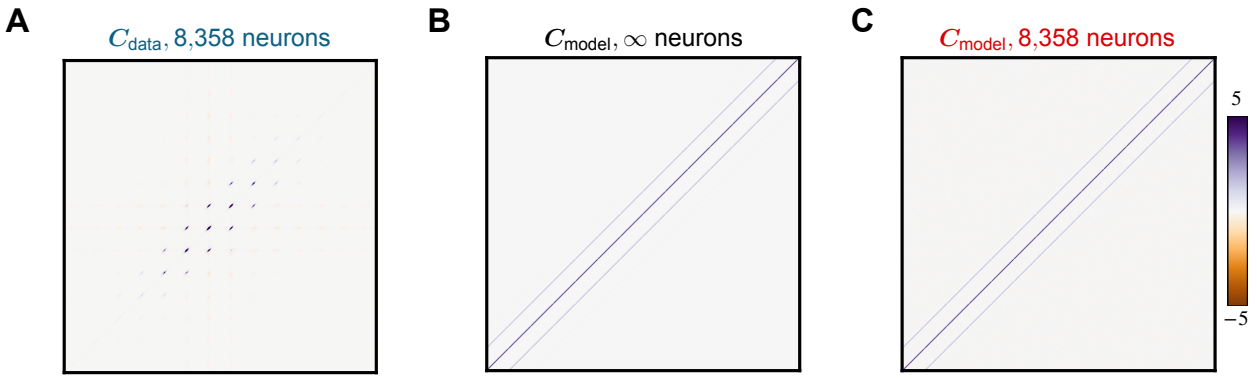

Figure B.9: **Covariance matrix of V1 receptive fields and translation invariant V1 model (Eq. B.22) for white noise stimuli.** We compare the covariance matrices generated from the (A) receptive fields of 8,358 mice V1 neurons, (B) translation invariant version of the V1 GP model and (C) 8,358 random samples from the model.

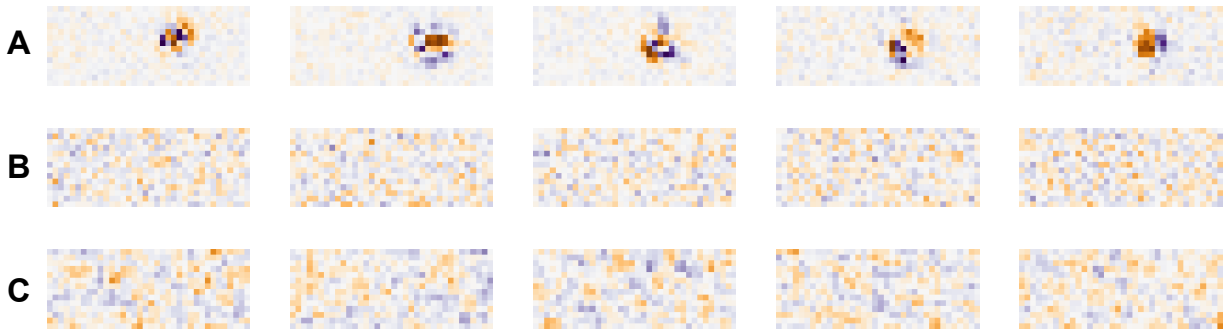

Figure B.10: **Receptive fields from V1 neurons, the unstructured model and the translation invariant V1 model (B.22).** We show the receptive fields from the (A) V1 neurons, (B) unstructured GP model and (C) the translation invariant V1 GP model.

Solving for the unknown constants leads to the equations

$$c_1 = \alpha + \delta - \frac{\alpha^2}{X} \qquad \Longrightarrow \; c_1 = \sqrt{\delta(2\alpha+\delta)},$$

$$c_2 = \frac{c_2\alpha}{X\left(1-\frac{c_2^2}{X}\right)^{1/2}} \qquad \Longrightarrow \; c_2 = \sqrt{(\alpha+\delta+c_1)\left(1-\frac{\alpha^2}{(\alpha+\delta+c_1)}\right)} = \sqrt{2c_1}.$$

The last step is to find the normalization constant for the eigenfunctions:

$$\begin{aligned}\int_{-\infty}^{\infty} |\hat{\phi}_k(t)|^2 dt &= \int_{\infty}^{\infty} e^{-2c_1t^2} H_k^2(c_2t)dt \\ &= \frac{1}{c_2}\int_{\infty}^{\infty} e^{-u^2} H_k^2(u)du \qquad (u = c_2t,\, c_2^2 = 2c_1) \\ &= \frac{2^k k!\sqrt{\pi}}{c_2}. \qquad ([92],\, 7.374.1)\end{aligned}$$

Therefore, our orthonormal eigenfunctions and eigenvalues for the 1-dimensional covariance are

$$\phi_k(t) = \frac{c_2}{2^k k!\sqrt{\pi}} e^{-c_1t^2} H_k(c_2t), \qquad \lambda_k^2 = \sqrt{\frac{\pi}{\alpha+\delta+c_1}}\left(1-\frac{c_2^2}{\alpha+\delta+c_1}\right)^{k/2}, \tag{B.23}$$

where $c_1 = \sqrt{\delta(2\alpha+\delta)}, c_2 = \sqrt{2c_1}$. Note that $\lambda_k^2 \propto c_3^k$ with $c_3 = \sqrt{1-\frac{c^2}{\alpha+\delta+c_1}}$, so that the spectrum decays exponentially.

B.7 Distributed receptive field centers imply a sum kernel space

To generate our V1-inspired weights, we first sample a center $\mathbf{c}$ uniformly at random from the pixels in the image; call this set of pixels S. We will now derive the kernel for this weight sampling.

Suppose that all of the weights are sampled with a single center $\mathbf{c}$. Then Eq (B.4) tells us that the structured kernel associated with the RFN

$$\begin{aligned} k_{\text{struct}}(\mathbf{x}, \mathbf{x}'; \mathbf{c}) &= k_{\text{unstruct}}(\mathbf{\Lambda_c \Phi_c}^T \mathbf{x}, \mathbf{\Lambda_c \Phi_c}^T \mathbf{x}') \\ &= k_{\text{unstruct}}(\tilde{\mathbf{x}}_\mathbf{c}, \tilde{\mathbf{x}}'_\mathbf{c}), \end{aligned} \tag{B.24}$$

where we have defined the *local basis change*

$$\tilde{\mathbf{x}}_\mathbf{c} = \mathbf{\Lambda_c \Phi_c}^T \mathbf{x}. \tag{B.25}$$

This local basis change projects into a basis of Hermite wavelets $\mathbf{\Phi_c}$ centered at $\mathbf{c}$ and filters according to the eigenvalues $\mathbf{\Lambda_c}$. The reproducing kernel Eq (B.24) defines an RKHS of functions $\mathcal{H}_\mathbf{c}$ which take images as their input and produce a real-valued output. The RKHS is a Hilbert space and thus has a norm $\|\cdot\|_{\mathcal{H}_\mathbf{c}}$. Functions with small $\mathcal{H}_\mathbf{c}$-norm are, informally, smooth functions of the local wavelet coefficients $\tilde{\mathbf{x}}_\mathbf{c}$.

In our experiments, we actually sample weights from all centers $\mathbf{c} \in S$ with equal probability. Taking the expectation over the centers, this means that the kernel will be an average over all of the local kernels Eq (B.24),

$$k_{\text{struct}}(\mathbf{x}, \mathbf{x}') = \frac{1}{|S|} \sum_{\mathbf{c} \in S} k_{\text{struct}}(\mathbf{x}, \mathbf{x}'; \mathbf{c}). \tag{B.26}$$

Let $\mathcal{H}$ be the RKHS associated with $k_{\text{struct}}(\cdot, \cdot)$, another space of functions that take in images and output a real number. The sum Eq (B.26) implies that $\mathcal{H} = \bigoplus_{\mathbf{c} \in S} \mathcal{H}_\mathbf{c}$, i.e. the RKHS is a direct sum of local RKHS's [197]. This means that any function $f \in \mathcal{H}$ can be written as $f = \sum_{\mathbf{c} \in S} f_\mathbf{c}$, with every $f_\mathbf{c} \in \mathcal{H}_\mathbf{c}$. The norm of this function comes from taking a minimum over all such decompositions

$$\|f\|_\mathcal{H} = \min_{f_\mathbf{c}: f = \sum_{\mathbf{c} \in S} f_\mathbf{c}} \sqrt{|S| \sum_{\mathbf{c} \in S} \|f_\mathbf{c}\|^2_{\mathcal{H}_\mathbf{c}}}.$$

We can think of functions with small $\mathcal{H}$-norm, which will be easiest to learn, as sums of smooth functions of local wavelet coefficients. This is equivalent to using convolutions, i.e. [144].

B.8 Timeseries data generation

We detail how the two frequency classification tasks from (detection and XOR) are generated. In both tasks, each example is an L ms timeseries sampled at f Hz, making each $\mathbf{x}$ a vector of length $d = L \times f$. Thus in the discrete setting, we only have d total frequencies. While the math below show continuous signals, in our code we generate analagous discrete signals using the discrete Fourier transform basis.

Frequency detection

The frequency detection task is a binary classification task. The positive examples contain a pure sinusoidal signal with frequency f_1 and additive Gaussian noise. The negative examples are just white noise. They are generated in the following way:

$$x_+(t) = \overbrace{a\sqrt{\frac{2}{L}}\left(\eta_f \cos(f_1 t) - \xi_f \sin(f_1 t)\right)}^{\text{pure frequency}} + \overbrace{\sqrt{1-a^2}\sqrt{\frac{2}{(d-1)L}} \sum_{\substack{j=0 \\ j:\omega_j \neq f_1}}^{d-1} \left(\eta_j \cos(\omega_j t) - \xi_j \sin(\omega_j t)\right)}^{\text{additive Gaussian noise}}$$

$$x_-(t) = \sqrt{\frac{2}{dL}} \sum_{j=0}^{d-1} \left(\eta_j \cos(\omega_j t) - \xi_j \sin(\omega_j t)\right)$$

where $\omega_j = 2\pi j/L$ is the j-th natural frequency, a is a parameter that sets the SNR, and the coefficients $\eta_j, \xi_j, \eta_f, \xi_f$ are random variables uniformly sampled from the unit circle (which gives each frequency component a random phase).

We define

$$\text{SNR} := \frac{a^2}{1-a^2}, \tag{B.27}$$

with $a \in [0, 1]$. Larger a means a larger contribution of the pure tone and smaller amplitude noise. Note that $\|x_-(t)\|^2_{L^2([0,L])} = \|x_+(t)\|^2_{L^2([0,L])} = 1$. The generation process ensures that the L^2 energy of both the negative and positive examples are matched and that the SNR is equal to the ratio of energy captured in frequency f_{lo} to the total energy in all other components.

We generate a balanced dataset with 7,000 timeseries signals which we split into a training set with 5,600 examples and a test set with 1,400 examples. We tuned the GP covariance parameters

f_{lo}, f_{hi}, and γ from

$$C(t,t') = \overbrace{\exp\left(-\frac{(t+t')}{\gamma}\right)}^{\text{localized}} \overbrace{\sum_{k=0}^{\infty} \lambda_k^2 \cos\left(\omega_k(t-t')\right)}^{\text{stationary process}}, \qquad \lambda_k = \overbrace{\begin{cases} 1 & f_{\text{lo}} \leq \omega_k \leq f_{\text{hi}} \\ 0 & \text{otherwise} \end{cases}}^{\text{bandlimited, flat-power spectrum}}, \quad \text{(B.28)}$$

using 3-fold cross validation on the training set. We found the optimal parameters for a network with 20 hidden neurons and used them for all hidden layer widths. We tested f_{lo} and f_{hi} parameters from 10 Hz to 200 Hz at increments of 10 Hz. For γ, we set the parameter range to be from 10 ms to 100 ms and used all parameters at increments of 10 ms. Using grid search, we tested all combinations of these parameters. The optimal model was refit using all training 5,600 samples, and the errors we report were measured on the test set.

Frequency XOR

We use a similar set up to generate the timeseries for the frequency exclusive-or (XOR) task. The positive examples are either frequency f_1 or f_2 Hz pure sinusoids with additive Gaussian noise. The negative examples are either mixed frequency timeseries (with both f_1 and f_2 Hz signals) or pure Gaussian noise. They are generated in the following way:

$$x_{+,k}(t) = \overbrace{a\sqrt{\frac{2}{L}}\left(\eta_f \cos(f_k t) - \xi_f \sin(f_k t)\right)}^{\text{pure frequency}} + \overbrace{\sqrt{\frac{2(1-a^2)}{(d-1)L}} \sum_{\substack{j=0 \\ j:\omega_j \neq f_j}}^{d-1} \left(\eta_j \cos(\omega_j t) - \xi_j \sin(\omega_j t)\right)}^{\text{additive Gaussian noise}}$$

$$x_{-,\text{noise}}(t) = \sqrt{\frac{2}{dL}} \sum_{j=0}^{d-1} \left(\eta_j \cos(\omega_j t) - \xi_j \sin(\omega_j t)\right)$$

$$x_{-,\text{mixed}}(t) = \frac{a}{\sqrt{L}} \sum_{j \in \{1,2\}} \left(\eta_j \cos(f_j t) - \xi_j \sin(f_j t)\right) + \sqrt{\frac{2(1-a^2)}{(d-2)L}} \sum_{\substack{j=0 \\ j:\omega_j \neq f_1, f_2}}^{d-1} \left(\eta_j \cos(\omega_j t) - \xi_j \sin(\omega_j t)\right),$$

for $k \in \{1,2\}$ in the $x_{+,k}(t)$ function. The constants, random variables, and details of SNR are identical to the frequency detection section. The datasets we generate have balanced proportions of $x_{+,1}$, $x_{+,2}$, $x_{-,\text{noise}}$, and $x_{-,\text{mixed}}$ signals.

B.9 SNR amplification via filtering

Let's consider the simple frequency detection task with stationary bandpass features. Since these features are stationary, their eigenvectors are Fourier modes, i.e. $\mathbf{\Phi}$ is the discrete Fourier transform (DFT) matrix. Assume the features encode a bandpass filter, which means that $\mathbf{\Lambda} = \text{diag}(\lambda_i)$, $i \in \{0, \ldots, d-1\}$, with $\lambda_i = 1$ for $i_{\text{lo}} \leq i \leq i_{\text{hi}}$ and 0 otherwise.

In frequency detection, a single frequency component (discrete Fourier mode) contains the signal with energy a^2. The other $d-1$ components each have energy $\frac{1-a^2}{d-1}$, for a total energy of $1-a^2$ contained in this white noise. After the basis change is applied to any input $\mathbf{x}$, the transformed vector $\tilde{\mathbf{x}} = \mathbf{\Lambda}\mathbf{\Phi}^*\mathbf{x}$ will have zeros in all entries outside the passband. (We use the conjugate transpose $\mathbf{\Phi}^*$ here rather than the transpose since the DFT matrix is complex; the interpretation is the same.) This makes the new representation $\tilde{\mathbf{x}}$ effectively d'-dimensional, where $d' = i_{\text{hi}} - i_{\text{lo}}$.

Now, first assume that the signal is within the passband. The total noise energy in the transformed representation becomes $(1-a^2)\frac{d'-1}{d-1}$, since one of the d' components is still signal, and no energy is lost in the retained components because $\mathbf{\Phi}$ is unitary. The overall noise is shrunk by a factor of $\frac{d'-1}{d-1}$, so the SNR gets boosted by $\frac{d-1}{d'-1}$. In the limiting case where $d' = 1$, the noise energy is 0 and SNR is infinite. On the other hand, if the signal lies outside the passband the SNR is reduced to 0.

B.10 Implementation details and code availability

In all experiments with RFNs, the training algorithm is an SVM classifier with squared hinge loss provided by the `sklearn.svm.LinearSVC` package and all other parameters set to their defaults. We used `scipy` [221] and `numpy` [98] to construct both classical unstructured and neural-inspired structured weights. For the experiments with fully-trained networks, we used `pytorch` [167]. The cross entropy loss function was optimized using full batch stochastic gradient gescent (SGD) optimizer, i.e. gradient descent (GD). Our code is available at `https://github.com/BruntonUWBio/structured-random-features`.

B.11 Covariance of V1 neurons with other stimuli

We repeat the covariance analysis of the V1-inspired weights on three additional datasets of V1 neurons. Different stimuli were shown in each dataset to calculate the receptive field.

The first dataset was provided by Ringach et al. from their work on characterizing the spatial structure of simple receptive fields in macaque (*Macaca fascicularis*) V1 [183]. The spikes of 250 neurons were recorded in response to drifting sinusoidal gratings. The receptive fields were calculated from the stimuli and responses using subspace reverse correlation. Because of the bandlimited properties of sinusoidal stimuli, this experiment biases the reconstruction towards smooth receptive fields. The receptive fields were of various sizes: 32 pixels $\times$ 32 pixels, 64 pixels $\times$ 64 pixels, and 128 pixels $\times$ 128 pixels. We resized them to a common dimension of 32 pixels $\times$ 32 pixels using local mean averaging. We find the optimal covariance parameters that fit the data to be $s = 2.41$ and $f = 0.95$ pixels. The covariance matrices and eigenfunctions are shown in Fig. B.11. Examples of biological receptive fields and random samples from the fitted model are shown in Fig. B.12 in the Appendix.

The second dataset contains the responses of 69,957 neurons recorded from the primary visual cortex of mice bred to express GCaMP6s. We presented 5,000 static natural images of 24 $\times$ 27 pixels in random order for 3 trials each. We calculated the receptive fields from the natural images and calcium responses of cells using ridge regression with an ℓ^2 penalty set to 0.1 after each image pixel was z-scored across images. We used the average receptive field over all three trials. For the covariance analysis, we picked cells with SNR > 0.4. This gave us 10,782 cells. The optimal covariance parameters that fit the data are $s = 5.40$ and $f = 1.17$ pixels. Examples of biological receptive fields and random samples from the model are shown in Fig. B.14. The covariance matrices and eigenfunctions are shown in Fig. B.13. Examples of biological receptive fields and random samples from the fitted model are shown in Fig. B.14. Repeating this analysis using receptive fields from individual trials yields identical results.

The third dataset contains the responses of 4,337 neurons also recorded from the primary visual cortex of mice bred to express GCaMP6s. The mice were shown static discrete Hartley transform (DHT, similar to a real-valued discrete Fourier transform) basis functions of size 30 $\times$ 80 pixels, and the calcium responses of neurons were recorded. The receptive fields were calculated using ridge

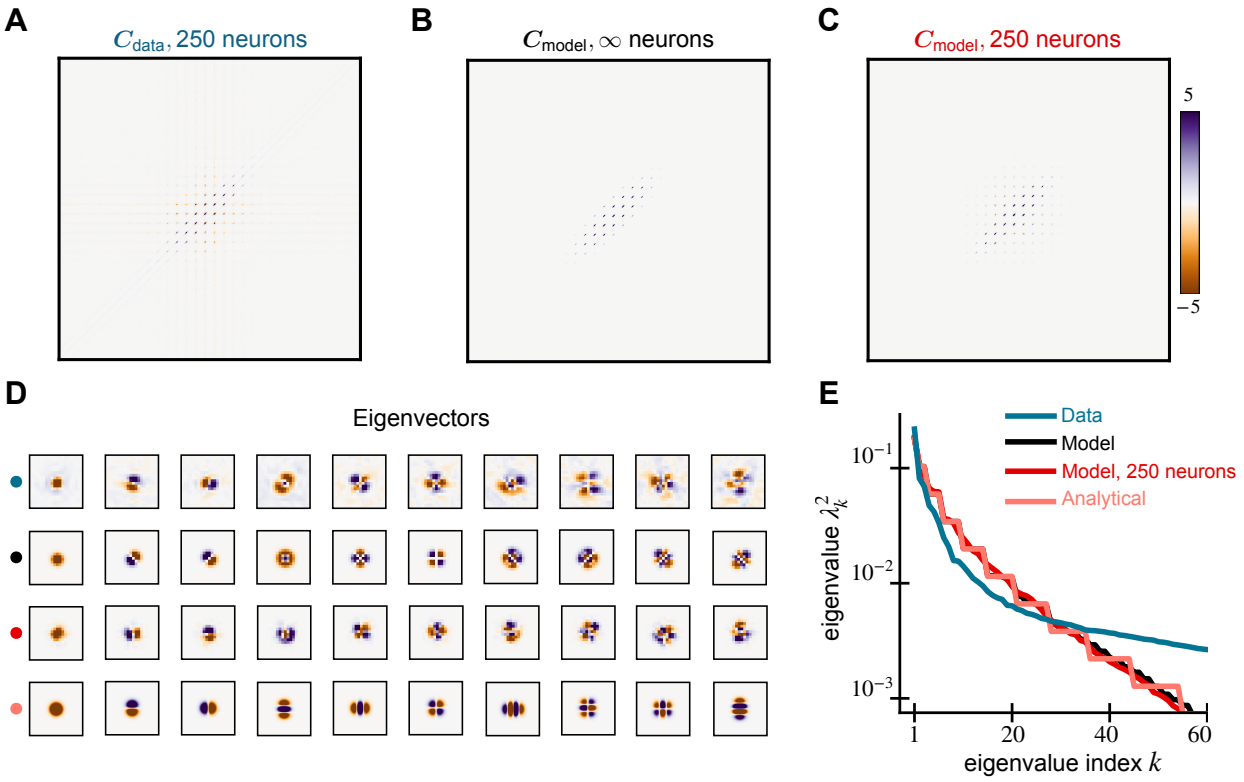

Figure B.11: **Spectral properties of V1 receptive fields and our model for Ringach dataset.** We compare the covariance matrices generated from the (A) receptive fields of 250 macaque V1 neurons, (B) the GP model Eq (B.21), and (C) 250 random samples from the model. The data is from [183]. (D) The leading 10 eigenvectors of the data and model covariance matrices show similar structure and explain 57% of the variance in the data. Analytical Hermite wavlet eigenfunctions are in the last row. (E) The eigenspectrum of the model matches well with the data.

regression without any ℓ^2 penalty. Here, we picked cells with SNR > 1 for analysis. We were left with 2,698 cells. The optimal covariance parameters that fit the data are $s = 10.46$ and $f = 1.20$ pixels. The covariance matrices and eigenfunctions are shown in Fig. B.15. Examples of biological receptive fields and random samples from the fitted model are shown in Fig. B.16.

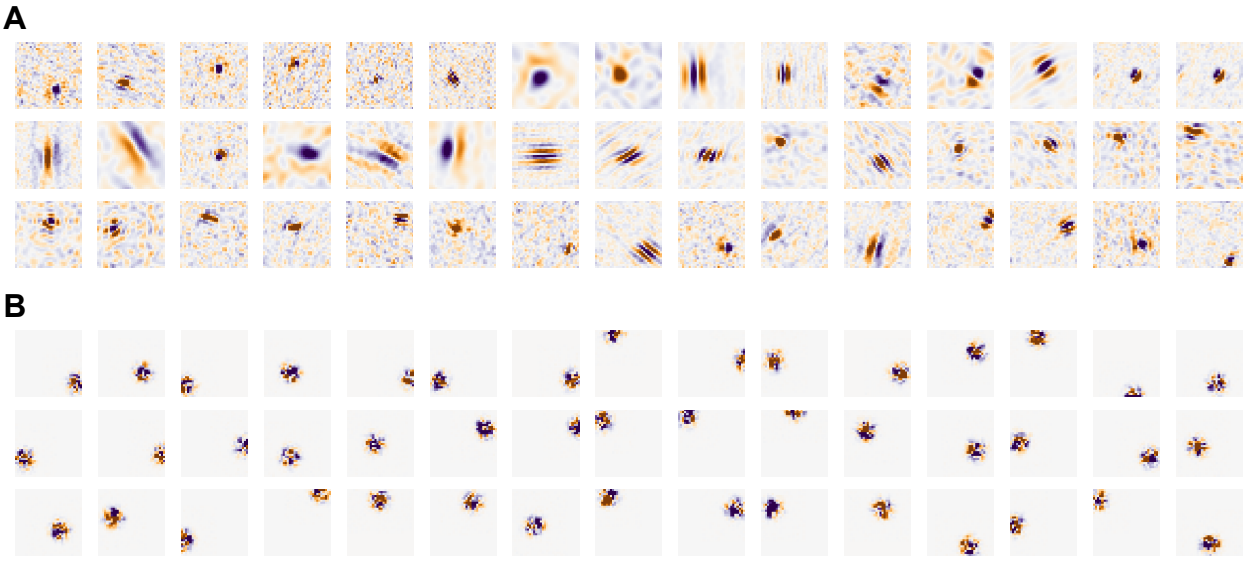

Figure B.12: **Receptive fields of V1 neurons from the Ringach dataset.** We show (A) biological receptive fields and (B) random samples from the fitted covariance model.

B.12 Initialization of networks with structured weights

We show results of initializing fully trained neural networks across a range of network widths (50, 100, 400, and 1,000) and learning rates (10^{-3}, 10^{-2}, and 10^{-1}) in Figures B.17, B.18, B.19, and B.20.

B.13 Deep network experiments

We experimented with using the V1-inspired weight initialization in the first two convolutional layers of AlexNet [126] and training on the ImageNet Large Scale Visual Recognition Challenge from 2012 [189]. Our implementation was based on the example provided by `pytorch` and `torchvision` [167] and used the same optimization routine, parameters, and schedule as in `https://github.com/pytorch/examples/tree/master/imagenet`.

All convolutional layers were initialized with weights drawn from a Gaussian distribution with variance $(c_{\text{in}} d_{\text{x}} d_{\text{y}})^{-1}$, where c_{in} was the number of input channels, and d_{x} and d_{y} are the dimensions

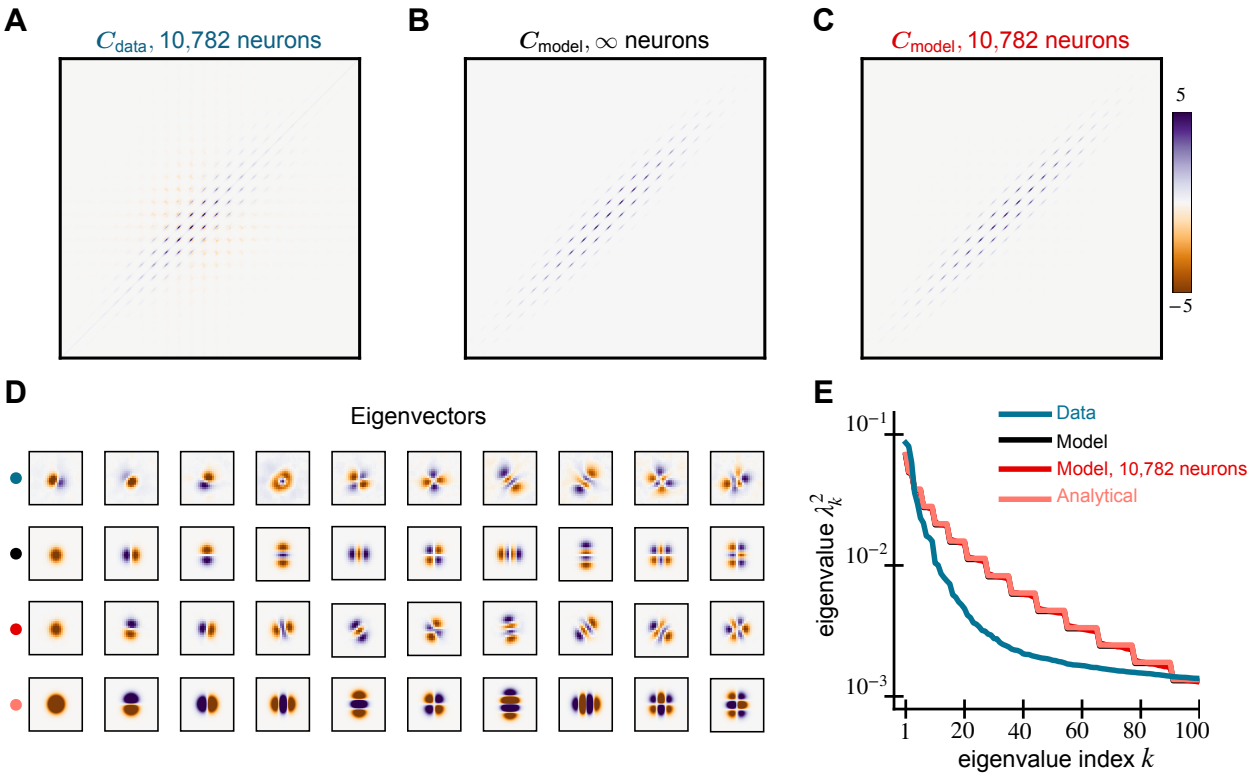

Figure B.13: **Spectral properties of V1 receptive fields and our model for natural image stimuli.** We compare the covariance matrices generated from the (A) receptive fields of 10,782 mice V1 neurons, (B) the GP model Eq (B.21), and (C) 10,782 random samples from the model. (D) The leading 10 eigenvectors of the data and model covariance matrices show similar structure and explain 39% of the variance in the data. Analytical Hermite wavelet eigenfunctions are in the last row. (E) The eigenspectrum of the model compared to the data.

of the filter. This is equal to the reciprocal of the fan-in. In the case of classical initialization, this Gaussian distribution has covariance proportional to the identity, whereas in the structured case we use the V1-inspired covariance centered in the center of the filter with independent draws for each input channel. All biases and weights in the other layers are set with their `pytorch` defaults.

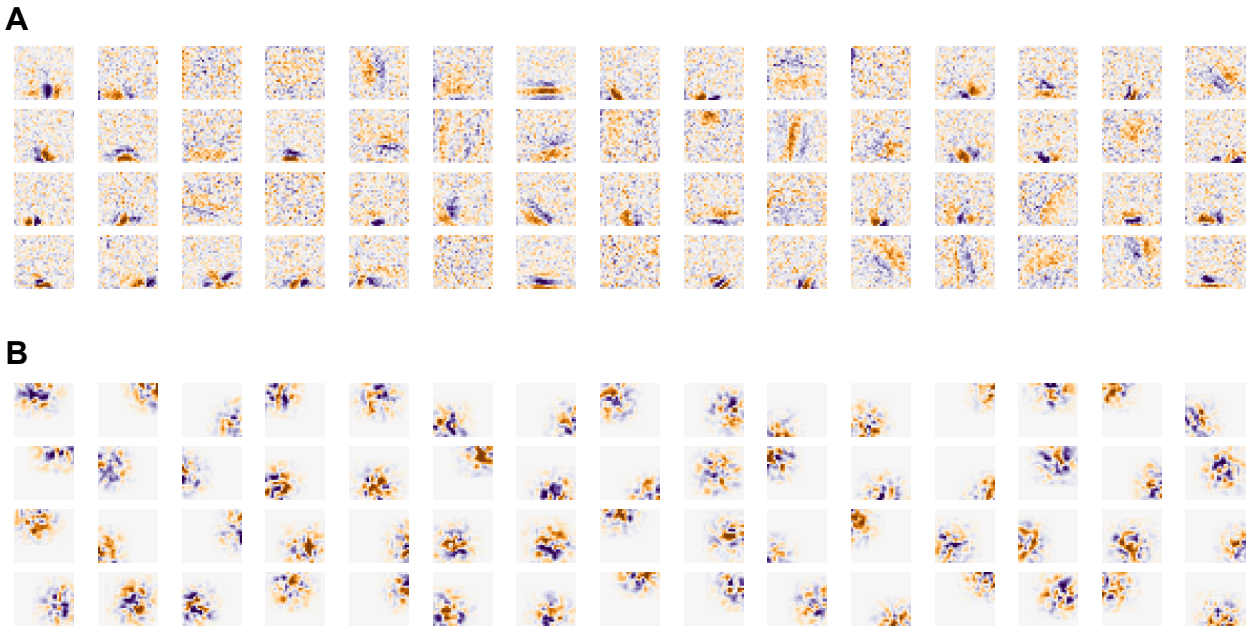

Figure B.14: **Receptive fields of V1 neurons from natural images stimuli.** We show (A) biological receptive fields and (B) random samples from the fitted covariance model.

The structured weights were only used in the first two convolutional layers of dimensions $d_x \times d_y =$ 11×11 and 5×5. The size parameter was set to $s = \max(d_x, d_y) \cdot 3$ and frequency bandwidth was $f = \max(d_x, d_y)/5$.

We show training and testing loss over the first 10 epochs for both the classical and structured initializations in Fig. B.21. The structured initialization at first shows an advantage over classical, with consistently lower losses for the first 4 epochs, but eventually the classical network catches up. From this point onwards (until the 90 training epochs are complete), the classical network has the same or lower loss. Both networks end up performing well, reaching accuracies close to those reported in [126] and the `torchvision` documentation (`https://pytorch.org/vision/stable/models.html`), as shown in Table B.3. The classical initialization performs slightly better overall.

These null results are perhaps not surprising: The initial layers of AlexNet contain only 64 and 192 output channels (i.e. filters) respectively, making up only a small fraction of the total weights

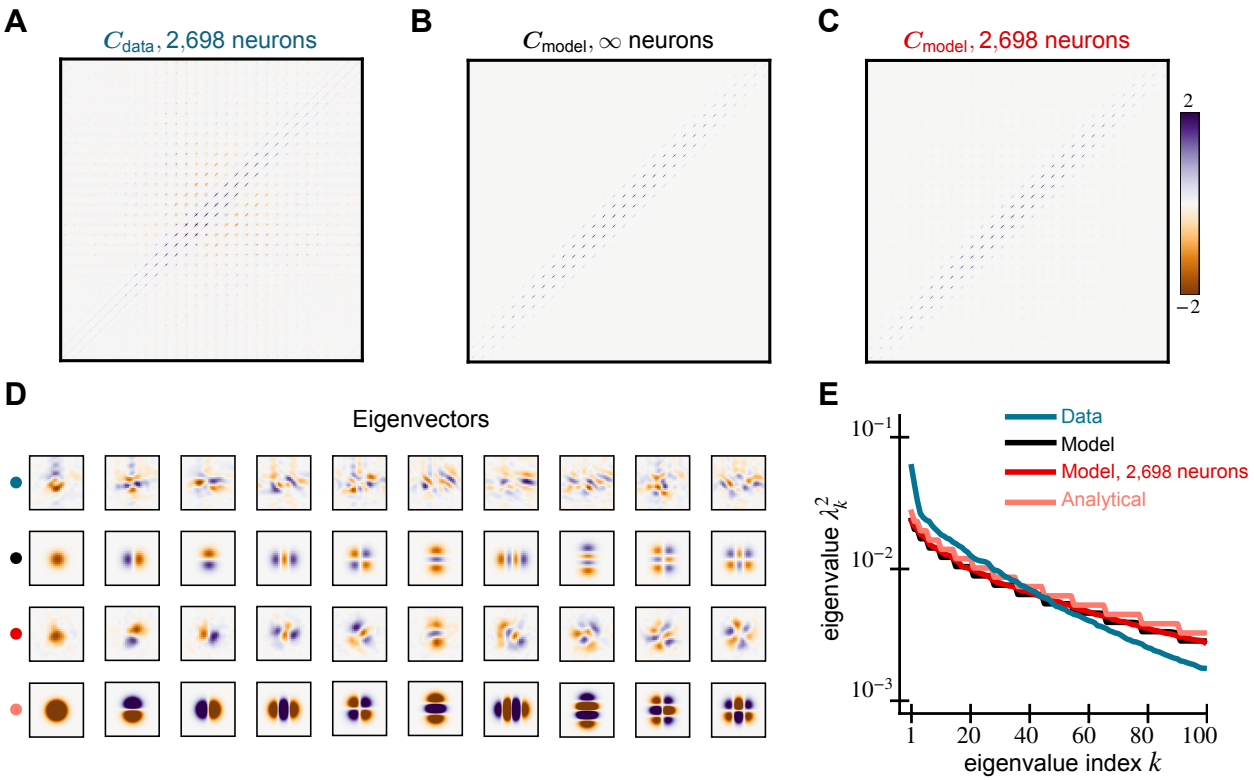

Figure B.15: **Spectral properties of V1 receptive fields and our model for DHT stimuli.** We compare the covariance matrices generated from the (A) receptive fields of 2,698 mice V1 neurons, (B) the GP model Eq (B.21), and (C) 2,698 random samples from the model. (D) The leading 10 eigenvectors of the data and model covariance matrices. They explain 29% of the variance in the data. Analytical Hermite wavelet eigenfunctions are in the last row. (E) The eigenspectrum of the model matches well with the data.

in the network. The deeper convolutional layers contain many more channels and are built with small 3×3 filters where our initialization is unlikely to help. It is also possible that the effects of initialization are less important for overparametrized models or with large amounts of training data.

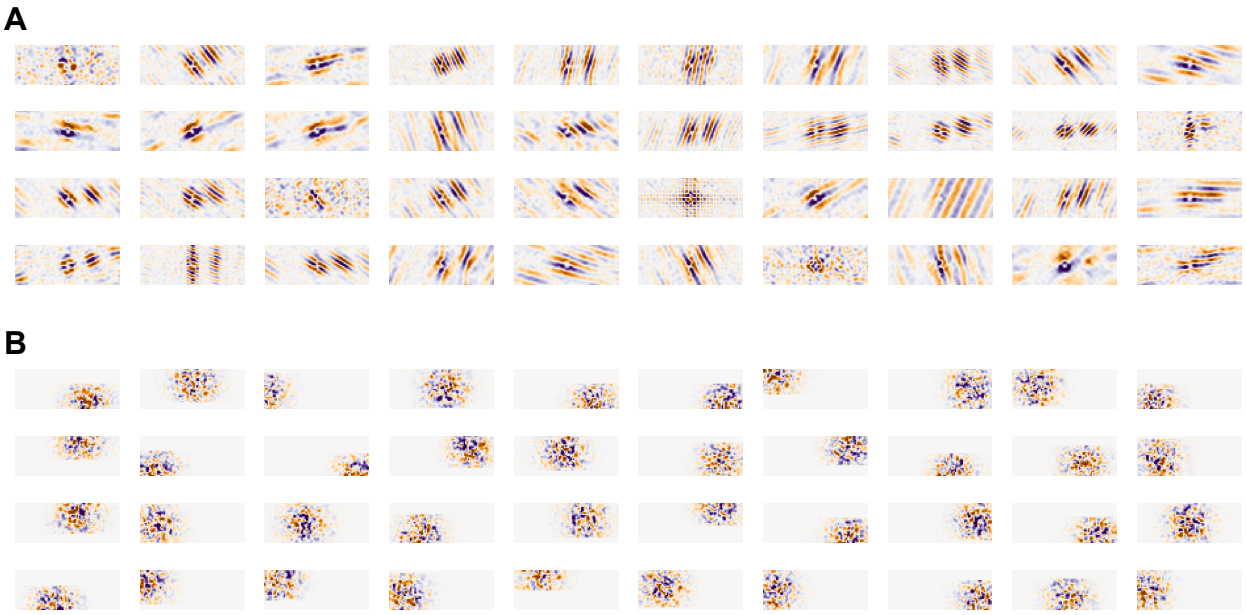

Figure B.16: **Receptive fields of V1 neurons from DHT stimuli.** We show (A) biological receptive fields and (B) random samples from the fitted covariance model.

Initialization	Loss	Top-1 accuracy (%)	Top-5 accuracy (%)
Classical	2.059 (1.907)	55.2 (56.5)	76.3 (79.2)
Structured	2.074 (1.920)	53.0 (56.4)	76.0 (79.0)

Table B.3: **Results after training AlexNet for 90 epochs on ImageNet.** The classical initialization leads to slightly smaller loss and higher accuracy over the structured initialization. Test values are shown in parentheses; these are better than the training values due to dropout.

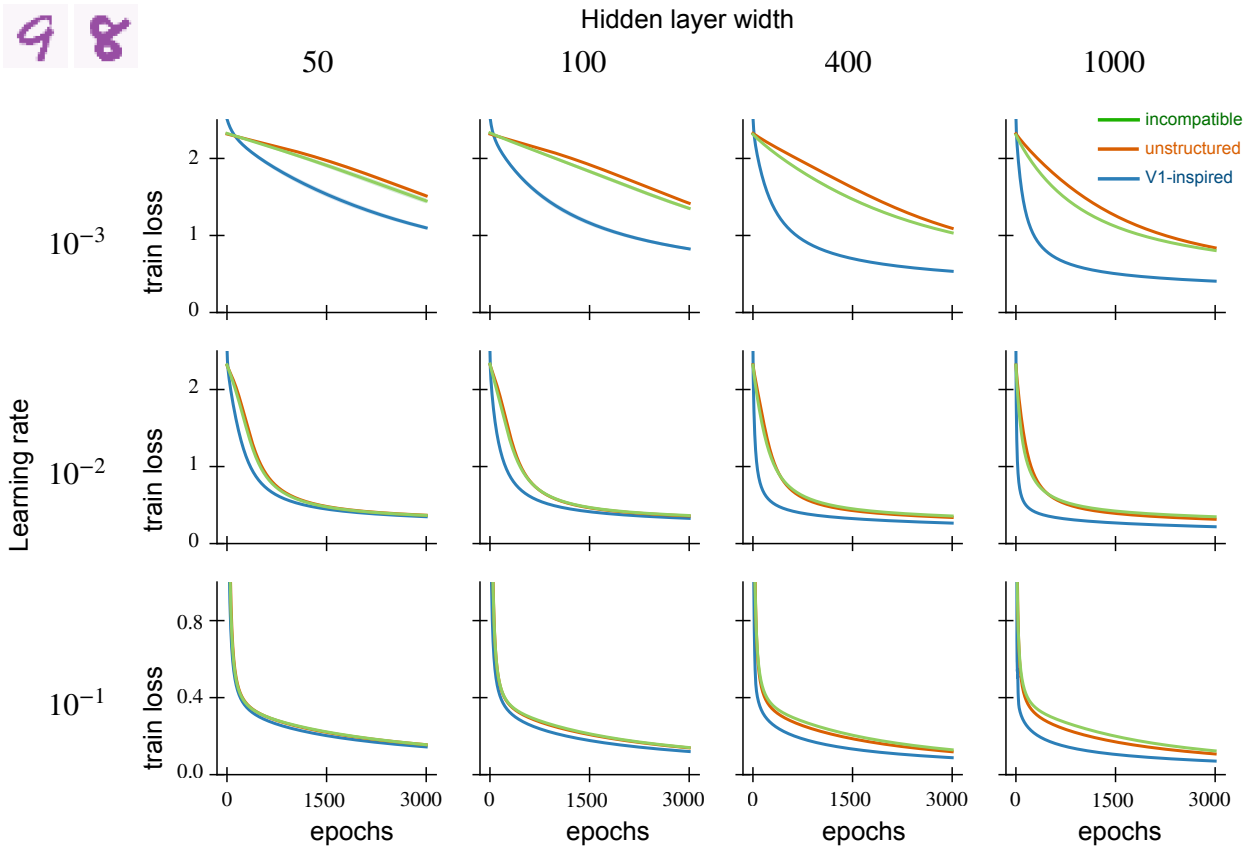

Figure B.17: **Training loss on MNIST for fully-trained neural networks initialized with V1 weights.** We show the average training loss of fully-trained networks against the number of training epochs across diverse hidden layer widths (50, 100, 400, and 1000) and learning rates ($10^{-1}, 10^{-2}$, and 10^{-3}). For every hidden layer width, we generate five random networks and average their performance. The solid lines show the average training loss while the shaded region represents the standard error. When the covariance parameters are tuned properly, V1-initialized networks achieve lower training loss over fewer epochs. The benefits are more significant at larger network widths and lower learning rates. With incompatible weights, V1 initialization leads to similar performance as unstructured initialization.

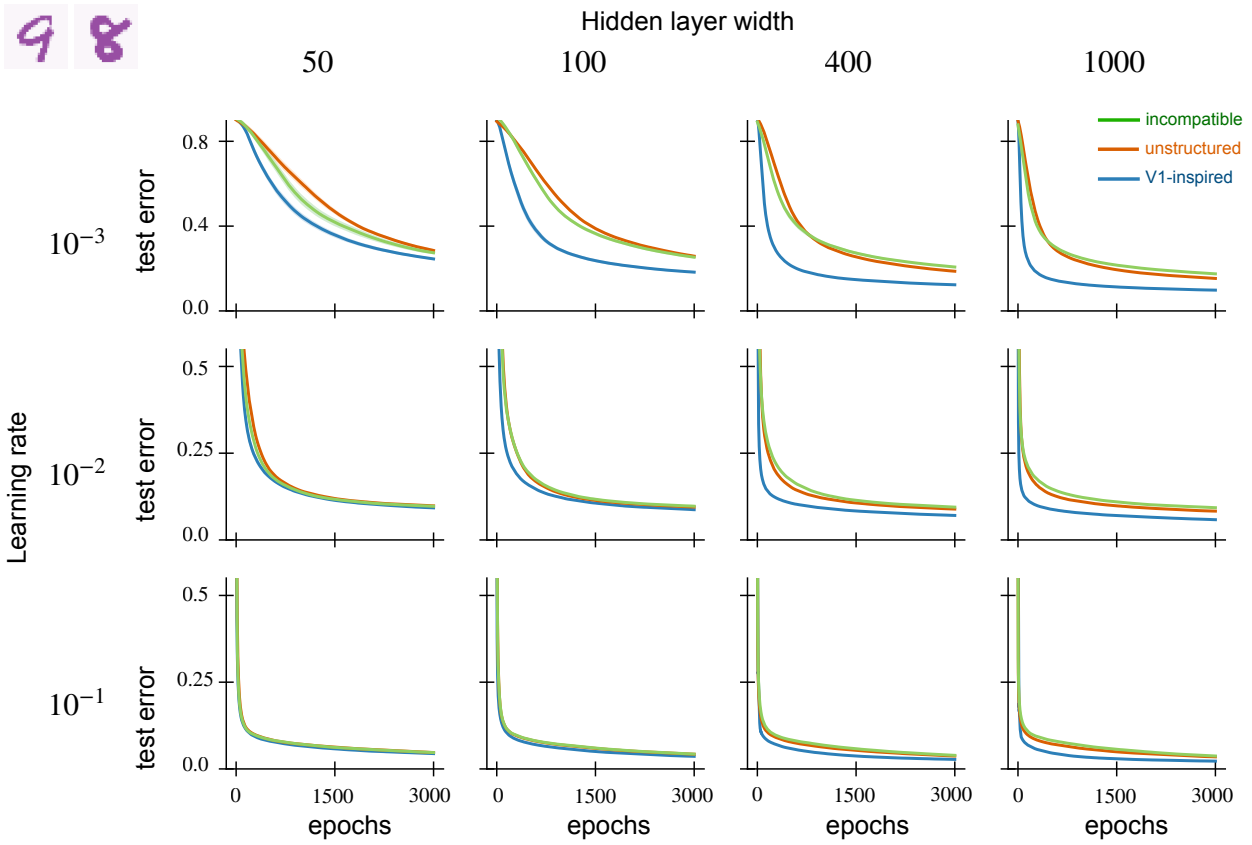

Figure B.18: **Test error on MNIST for fully-trained neural networks initialized with V1 weights.** We show the average test error of fully-trained networks against the number of training epochs across diverse hidden layer widths (50, 100, 400, and 1000) and learning rates ($10^{-1}, 10^{-2}$, and 10^{-3}). For every hidden layer width, we generate five random networks and average their performance. The solid lines show the average test error while the shaded regions represent the standard error. When the covariance parameters are tuned properly, V1-initialized networks achieve lower test error over fewer epochs. The benefits are more significant at larger network widths and lower learning rates. With incompatible weights, V1 initialization leads to similar performance as unstructured initialization.

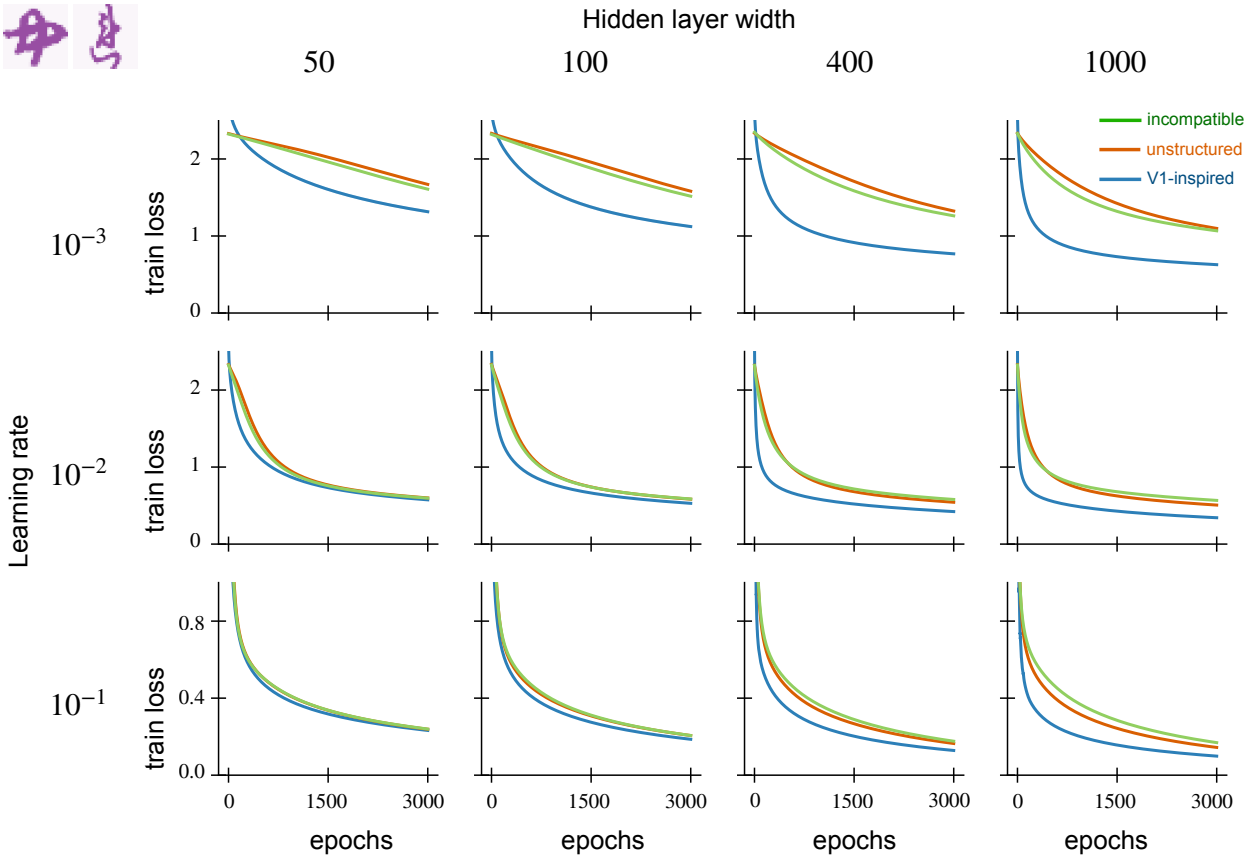

Figure B.19: **Training loss on KMNIST for fully-trained neural networks initialized with V1 weights.** We show the average training loss of fully-trained networks against the number of training epochs across diverse hidden layer widths (50, 100, 400, and 1000) and learning rates ($10^{-1}, 10^{-2}$, and 10^{-3}). For every hidden layer width, we generate five random networks and average their performance. The solid lines show the average training loss while the shaded regions represent the standard error. When the covariance parameters are tuned properly, V1-initialized networks achieve lower training loss over fewer epochs. The benefits are more significant at larger network widths and lower learning rates. With incompatible weights, V1 initialization leads to similar performance as unstructured initialization.

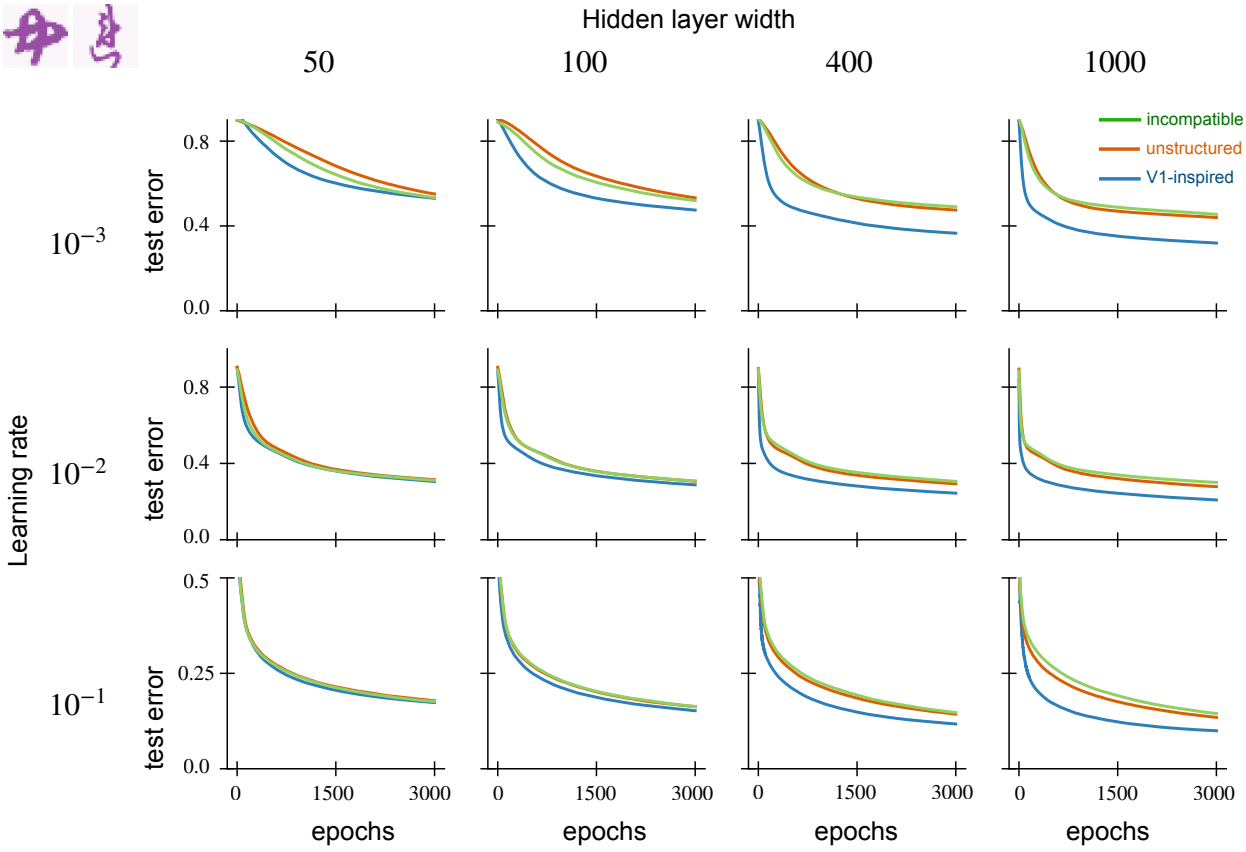

Figure B.20: **Test error on KMNIST for fully-trained neural networks initialized with V1 weights.** We show the average test error of fully-trained networks against the number of training epochs across diverse hidden layer widths (50, 100, 400, and 1000) and learning rates ($10^{-1}, 10^{-2}$, and 10^{-3}). For every hidden layer width, we generate five random networks and average their performance. The solid lines show the average test error while the shaded regions represent the standard error. When the covariance parameters are tuned properly, V1-initialized networks achieve lower test error over fewer epochs. The benefits are more significant at larger network widths and lower learning rates. With incompatible weights, V1 initialization leads to similar performance as unstructured initialization.

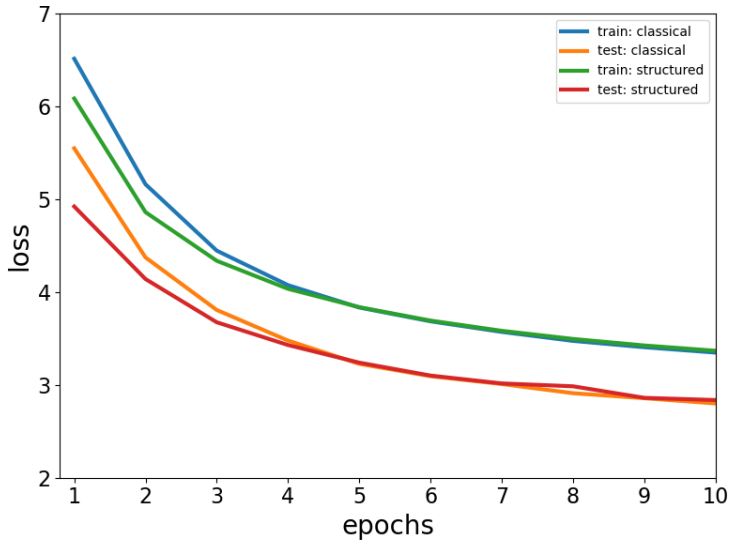

Figure B.21: **Initializing AlexNet using structured random features shows little benefit for ImageNet.** Training and testing loss are shown for classical and structured random initializations of convolutional layers in AlexNet. These losses are initially lower for structured features, but by 6 epochs the classical initialization catches up and it eventually reaches a slightly lower loss than the structured initialization. Note that the training losses are higher than testing due to dropout applied in the training phase.