Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall et al.,

University of California Berkeley, Google Research

NeurIPS 2020 Spotlight

Presenter: Minho Park

Demo

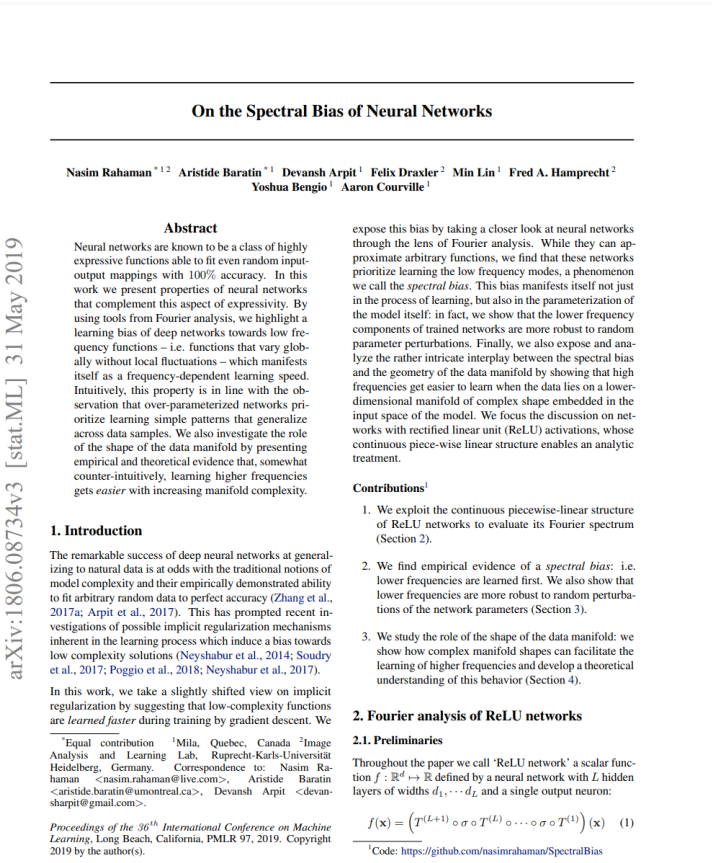
Figure 1: Fourier features improve the results of coordinate-based MLPs for a variety of high-frequency low-dimensional regression tasks, both with direct (b, c) and indirect (d, e) supervision. We visualize an example MLP (a) for an image regression task (b), where the input to the network is a pixel coordinate and the output is that pixel's color. Passing coordinates directly into the network (top) produces blurry images, whereas preprocessing the input with a Fourier feature mapping (bottom) enables the MLP to represent higher frequency details.

Demo

- https://bmild.github.io/fourfeat/img/lion_none_gauss_v1.mp4

On the Spectral Bias of Neural Networks

- Fourier analysis of ReLU networks
- Empirical evidence of a spectral bias



On the Spectral Bias of Neural Networks

Nasim Rahaman [*1 2] Aristide Baratin [*1] Devansh Arpit [1] Felix Draxler [2] Min Lin [1] Fred A. Hamprecht [2]
Yoshua Bengio [1] Aaron Courville [1]

Abstract

Neural networks are known to be a class of highly expressive functions able to fit even random input-output mappings with 100% accuracy. In this work we present properties of neural networks that complement this aspect of expressivity. By using tools from Fourier analysis, we highlight a learning bias of deep networks towards low frequency functions – i.e. functions that vary globally without local fluctuations – which manifests itself as a frequency-dependent learning speed. Intuitively, this property is in line with the observation that over-parameterized networks prioritize learning simple patterns that generalize across data samples. We also investigate the role of the shape of the data manifold by presenting empirical and theoretical evidence that, somewhat counter-intuitively, learning higher frequencies gets *easier* with increasing manifold complexity. We expose this bias by taking a closer look at neural networks through the lens of Fourier analysis. While they can approximate arbitrary functions, we find that these networks prioritize learning the low frequency modes, a phenomenon we call the *spectral bias*. This bias manifests itself not just in the process of learning, but also in the parameterization of the model itself: in fact, we show that the lower frequency components of trained networks are more robust to random parameter perturbations. Finally, we also expose and analyze the rather intricate interplay between the spectral bias and the geometry of the data manifold by showing that high frequencies get easier to learn when the data lies on a lower-dimensional manifold of complex shape embedded in the input space of the model. We focus the discussion on networks with rectified linear unit (ReLU) activations, whose continuous piece-wise linear structure enables an analytic treatment.

1. Introduction

The remarkable success of deep neural networks at generalizing to natural data is at odds with the traditional notions of model complexity and their empirically demonstrated ability to fit arbitrary random data to perfect accuracy (Zhang et al., 2017a; Arpit et al., 2017). This has prompted recent investigations of possible implicit regularization mechanisms inherent in the learning process which induce a bias towards low complexity solutions (Neyshabur et al., 2014; Soudry et al., 2017; Poggio et al., 2018; Neyshabur et al., 2017).

In this work, we take a slightly shifted view on implicit regularization by suggesting that low-complexity functions are *learned faster* during training by gradient descent. We

Contributions[1]

1. We exploit the continuous piecewise-linear structure of ReLU networks to evaluate its Fourier spectrum (Section 2).
2. We find empirical evidence of a *spectral bias*: i.e. lower frequencies are learned first. We also show that lower frequencies are more robust to random perturbations of the network parameters (Section 3).
3. We study the role of the shape of the data manifold: we show how complex manifold shapes can facilitate the learning of higher frequencies and develop a theoretical understanding of this behavior (Section 4).

2. Fourier analysis of ReLU networks

2.1. Preliminaries

Throughout the paper we call 'ReLU network' a scalar function $f : \mathbb{R}^d \mapsto \mathbb{R}$ defined by a neural network with L hidden layers of widths $d_1, \cdots d_L$ and a single output neuron:

$$f(\mathbf{x}) = \left(T^{(L+1)} \circ \sigma \circ T^{(L)} \circ \cdots \circ \sigma \circ T^{(1)}\right)(\mathbf{x}) \quad (1)$$

[*]Equal contribution [1]Mila, Quebec, Canada [2]Image Analysis and Learning Lab, Ruprecht-Karls-Universität Heidelberg, Germany. Correspondence to: Nasim Rahaman <nasim.rahaman@live.com>, Aristide Baratin <aristide.baratin@umontreal.ca>, Devansh Arpit <devansharpit@gmail.com>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

[1]Code: https://github.com/nasimrahaman/SpectralBias

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

Preliminaries

- ReLU networks

$$f(\boldsymbol{x}) = \left(T^{(L+1)} \circ \sigma \circ \cdots \circ \sigma \circ T^{(1)}\right)(\boldsymbol{x})$$

- Where $T^{(k)}: \mathbb{R}^{d_{k-1}} \to \mathbb{R}^{d_k}$ is affine function and $\sigma(u)_i = \max(0, u_i)$ denotes the ReLU activation function

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

Preliminaries

- ReLU network are known to be continuous piece-wise linear (CPWL) functions
- Remarkably, the converse also holds: **every CPWL function can be repr esented by a ReLU network** (Arora et al., 2018, Theorem 2.1)

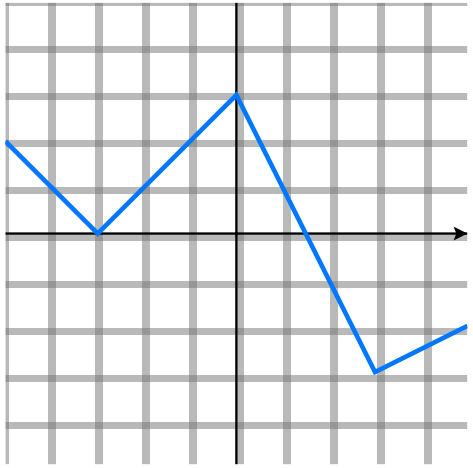


Figure 1. Continuous Piece-Wise Linear function

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734
Piecewise_linear_function, Wikipedia, https://en.wikipedia.org/wiki/Piecewise_linear_function
Understanding Deep Neural Networks with Rectified Linear Units, Raman Arora et al., ICLR 2018, https://openreview.net/pdf?id=B1J_rgWRW

Preliminaries

$$f(\boldsymbol{x}) = \sum_{\epsilon} 1_{P_\epsilon}(\boldsymbol{x})(W_\epsilon \boldsymbol{x} + b_\epsilon)$$

- Each region (P_ϵ) corresponds to an activation pattern of all hidden neurons of the network

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

Fourier Spectrum

- Structure of ReLU networks in terms of their Fourier representation

$$f(\boldsymbol{x}) = (2\pi)^{\frac{d}{2}} \int \tilde{f}(k) e^{ik\cdot x} dk$$

- Lemma 1. The Fourier transform of ReLU networks decomposes as,

$$\tilde{f}(k) = i \sum_{\epsilon} \frac{W_\epsilon k}{k^2} \tilde{1}_{P_\epsilon}(k)$$

- Simple Proof)
- $\int \nabla_x \cdot \left(k f(x) e^{-ik\cdot x}\right) dx = 0$
- $= \int \left(k \cdot (\nabla_x f)(x) - ik^2 f(x)\right) e^{-ik\cdot x} dx$

- $\tilde{f}(k) = \frac{1}{-ik^2} k \cdot \int (\nabla_x f)(x) e^{-ik\cdot x}\, dx$
- $(\nabla_x f)(x) = \sum_\epsilon 1_\epsilon W_\epsilon$

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

Fourier Spectrum

- Lemma 2. Let P be a full dimensional polytope in $\mathbb{R}^d$. Its Fourier spectrum takes the form:

$$\tilde{1}_P(k) = \int_P e^{-ik\cdot x} dx = \sum_{n=0}^{d} \frac{D_n(k) 1_{G_n}(k)}{k^n}$$

- where G_n is the union of n-dimensional subspaces that are orthogonal to some n-codimensional face of P, $D_n: \mathbb{R}^d \to \mathbb{C}$ is in $\Theta(1)$ $(k \to \infty)$

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

Fourier Spectrum

- Theorem 1. The Fourier components of the ReLU network f_θ with parameters θ is given by the rational function:

$$\tilde{f}_\theta(k) = \sum_{n=0}^{d} \frac{C_n(\theta, k) 1_{H_n^\theta}(k)}{k^{n+1}}$$

- Discussion: The spectral decay of ReLU networks is highly anisotropic in large dimensions.

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

Experiment 1

- Empirical evidence of a spectral bias: i.e. lower frequencies are learned fir st.
- Experiment 1. $\lambda: [0,1] \to \mathbb{R}$ given by

$$\lambda(z) = \sum_i A_i \sin(2\pi k_i z + \phi_i)$$

- A 6-layer deep 256-unit wide ReLU network f_θ is trained to regress λ with $\kappa = (5, 10, \ldots, 45, 50)$ and $N = 200$.

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

Experiment 1

- Results: Spectral norm $\left|\tilde{f}_\theta(k_i)\right|/A_i$

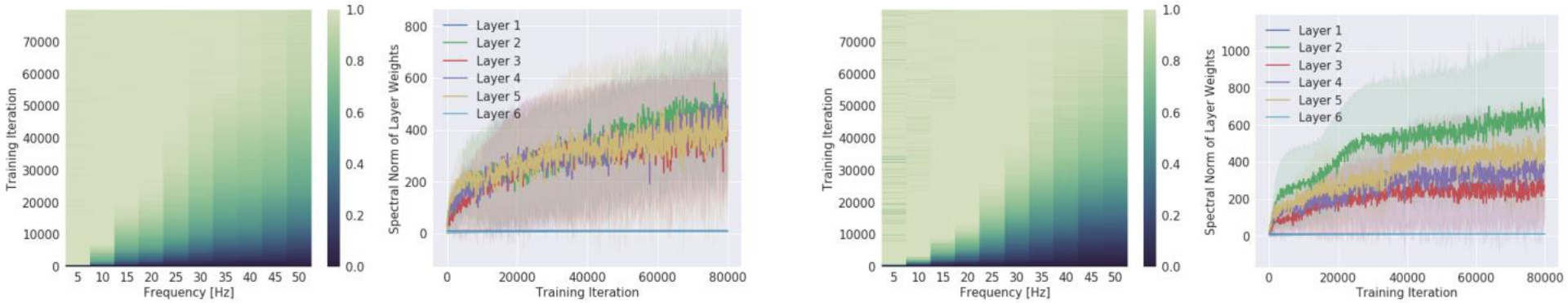


Figure 1. Left (a, b): Evolution of the spectrum (x-axis for frequency) during training (y-axis). Right (a, b) : Evolution of the spectral norm (y-axis) of each layer during training (x-axis). **Gist:** We find that even when higher frequencies have larger amplitudes, the model prioritizes learning lower frequencies first.

On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734

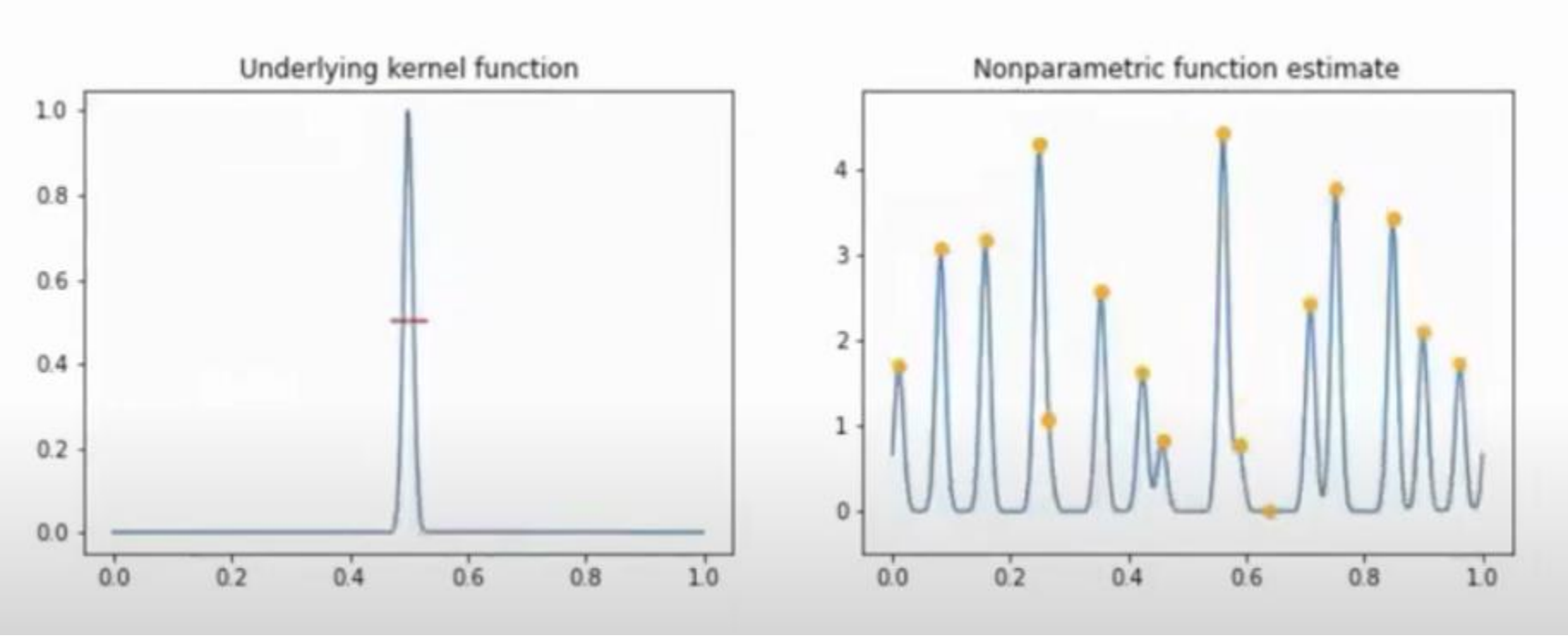
Linear Regression

- For training data points $\{(x_i, y_i)\}_{i=1}^{n}$,
- Linear estimation model $f(W, x) = W^T x$ is fitted for
- Minimize the loss function $\mathcal{L}(W) = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - f(W, x_i)\right)^2$
- Using the least squares optimization. $f(W, x) = \left((X^T X)^{-1} X^T y\right)^T x$

Kernel Regression

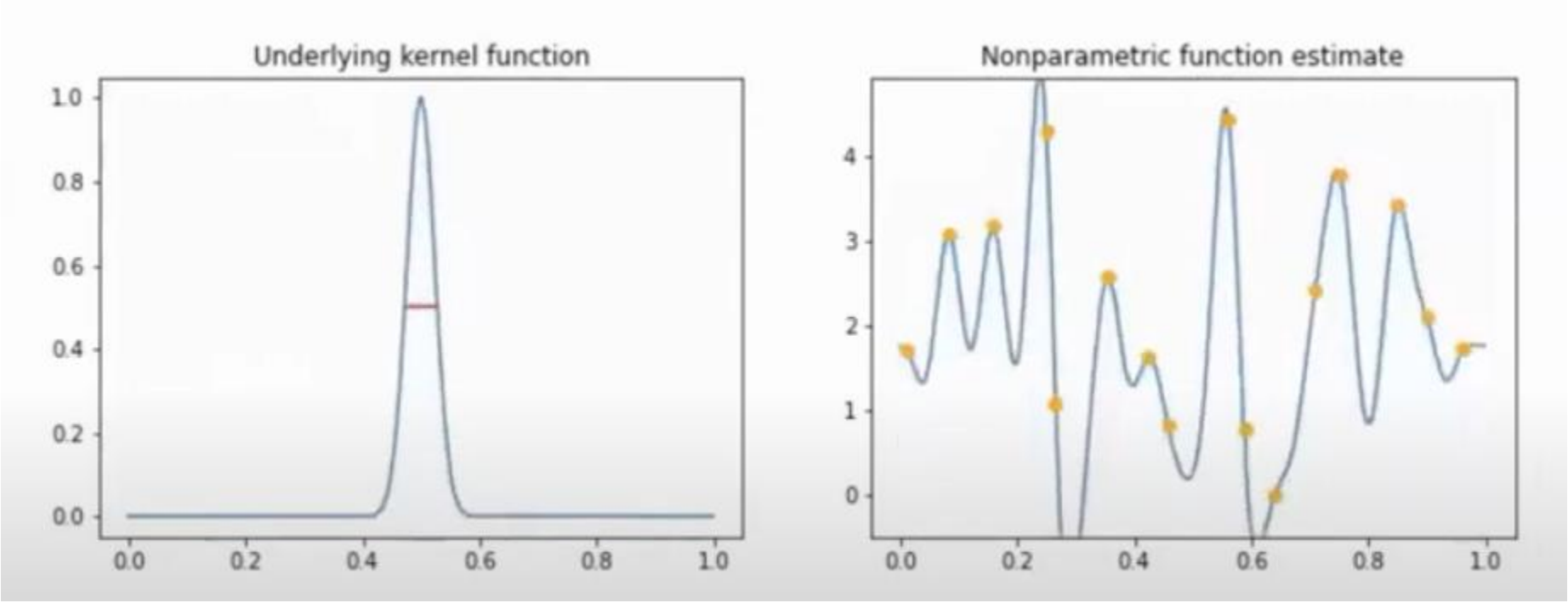
- For training data points $\{(x_i, y_i)\}_{i=1}^{n}$,
- With predefined kernel function ϕ: $\mathbb{R}^d \rightarrow \mathbb{R}^D$, D $\gg$ d
- Kernel estimation model $f(W, x) = \sum_{j=1}^{n} w_j \phi\left(x - x_j\right)$ is fitted for
- Minimize the loss function $\mathcal{L}(W) = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - f(W, x_i)\right)^2$
- Using the least squares optimization. $f(W, x) = \sum_{i=1}^{n}(K^{-1}y)_i k(x_i, x)$
  - $K_{ij} = k\left(x_i, x_j\right) = \phi^T(x_i)\phi\left(x_j\right)$

Width of Kernel



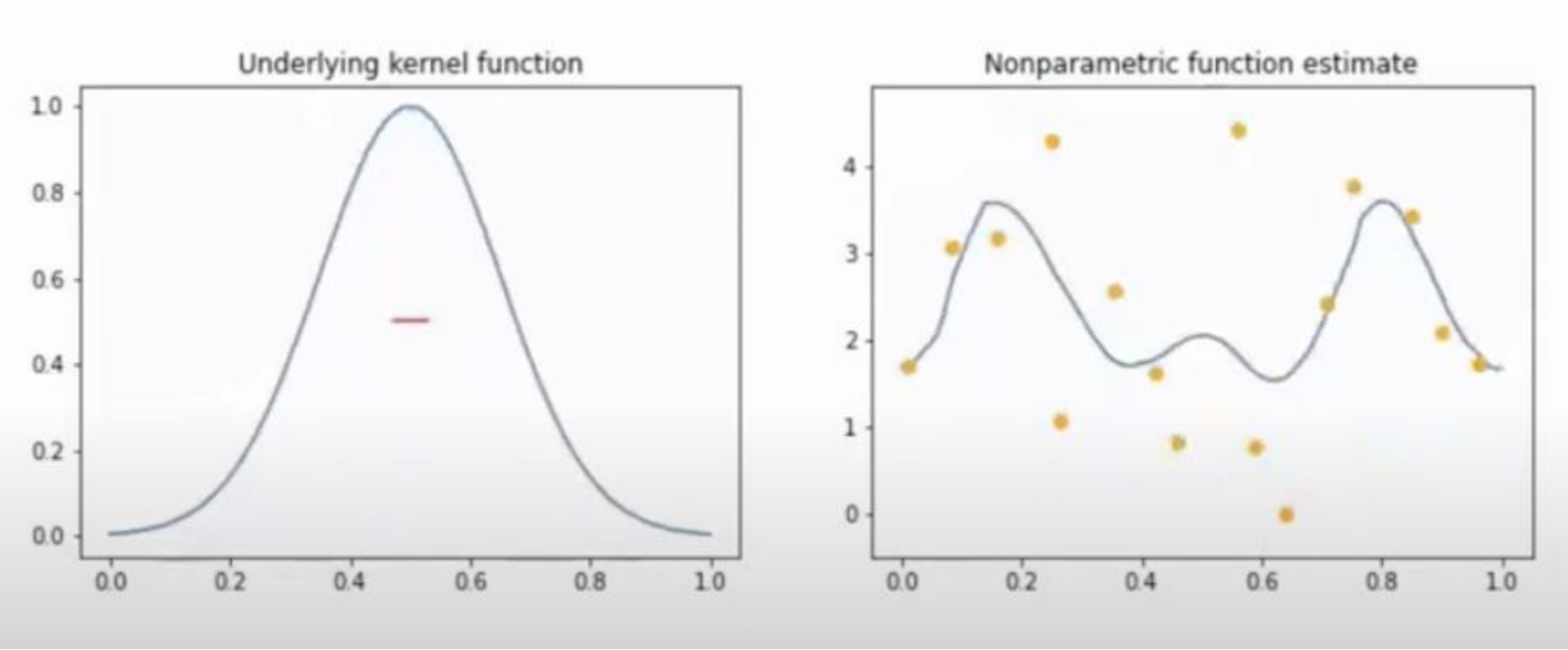
Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains, Matthew Tancik, https://youtu.be/h0SXP6lJxak

Width of Kernel



Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains, Matthew Tancik, https://youtu.be/h0SXP6lJxak

Width of Kernel

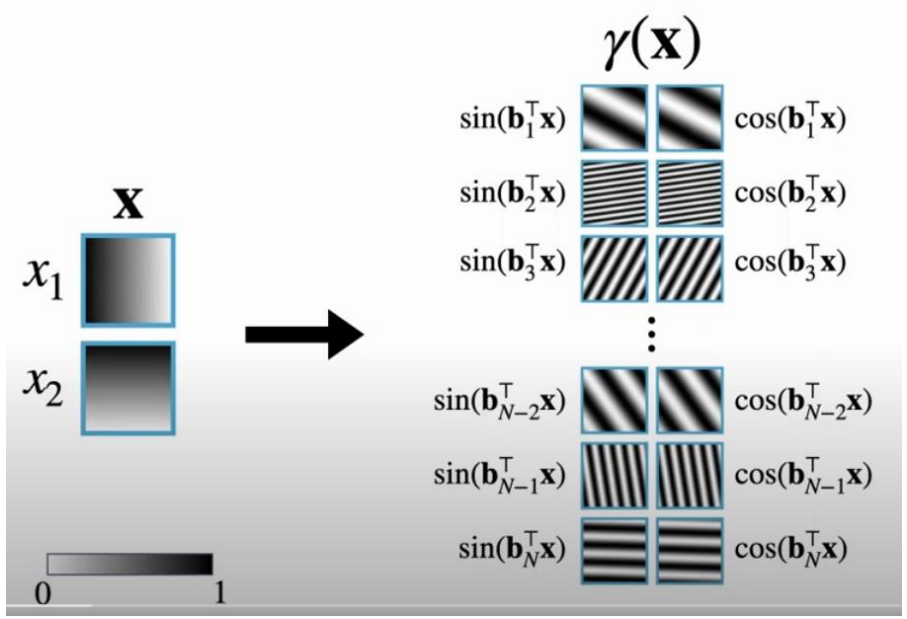


Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains, Matthew Tancik, https://youtu.be/h0SXP6lJxak

Fourier Feature Mapping

- Fourier feature mapping (a.k.a. Positional encoding)

$$\gamma(v) = [a_1 \cos(2\pi b_1^T v), a_1 \sin(2\pi b_1^T v), \cdots, a_m \cos(2\pi b_m^T v), a_m \sin(2\pi b_m^T v)]^T$$



Fourier feature mapping : simple 2D example

Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains, Matthew Tancik, https://youtu.be/h0SXP6lJxak

Neural Tangent Kernel

- Recent ML theory work shows that training neural network with gradient descent becomes the same as performing kernel regression as the width of each layer goes to infinity
- Using a Fourier feature mapping changes the corresponding kernel function (the NTK), allowing MLPs to represent higher frequency functions

Neural Tangent Kernel: Convergence and Generalization in Neural Networks, Arthur Jacot et al., https://arxiv.org/abs/1806.07572

Fourier Feature Mapping to NTKs

- Neural Tangent Kernel is a scalar function of dot product

$$\mathrm{NTK}(x, y) = h(x^T y)$$

- Dot product of Fourier feature mapping is simple:

$$\gamma(x)^T \gamma(y) = (\sin(Bx))^T \sin(By) + (\cos(Bx))^T \cos(By)$$

$$= \cos\big(B(x - y)\big)$$ Stationary

Fourier Feature Mapping to NTKs

- Fourier feature mapping lets us control width of Neural Tangent Kernel

$$\mathrm{NTK}\big(\gamma(x), \gamma(y)\big) = h(\cos B(x - y))$$

If we simply scale B, we can manipulate the width of the kernel

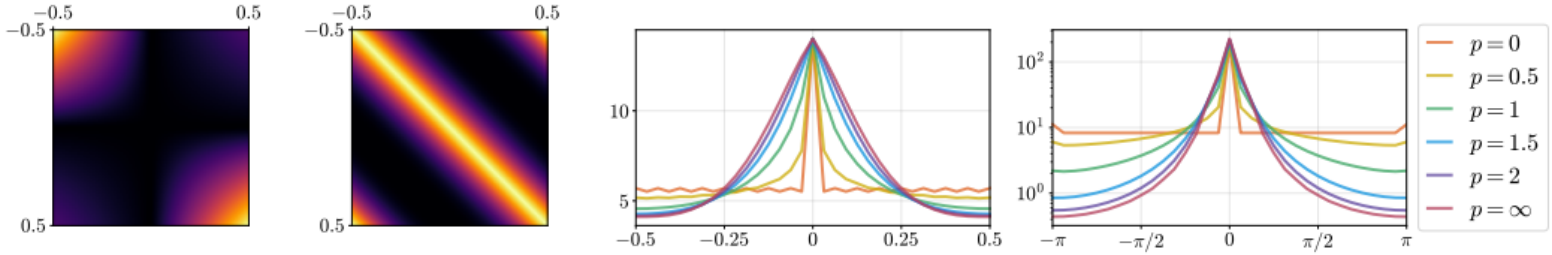
- Since B is sampled from random distribution, scale B using standard deviation of random distribution

Scaling B using standard deviation

- https://bmild.github.io/fourfeat/img/test_sweep_1e-4_5000_more_low.mp4

Manipulate the Fourier Feature Mapping

- p increase ⇒ Kernel width increase ⇒ Spectra width decrease



(a) No mapping NTK (b) Basic mapping NTK (c) NTK spatial (d) NTK Fourier spectrum

Figure 2: Adding a Fourier feature mapping can improve the poor conditioning of a coordinate-based MLP's neural tangent kernel (NTK). (a) We visualize the NTK function $k_{\text{NTK}}(x_i, x_j)$ (Eqn. 2) for a 4-layer ReLU MLP with one scalar input. This kernel is not shift-invariant and does not have a strong diagonal, making it poorly suited for kernel regression in low-dimensional problems. (b) A basic input mapping $\gamma(v) = [\cos 2\pi v, \sin 2\pi v]^{\mathrm{T}}$ makes the composed NTK $k_{\text{NTK}}(\gamma(v_i), \gamma(v_j))$ shift-invariant (stationary). (c) A Fourier feature input mapping (Eqn. 5) can be used to tune the composed kernel's width, where we set $a_j = 1/j^p$ and $b_j = j$ for $j = 1, \ldots, n/2$. (d) Higher frequency mappings (lower p) result in composed kernels with wider spectra, which enables faster convergence for high-frequency components (see Figure 3).

Manipulate the Fourier Feature Mapping

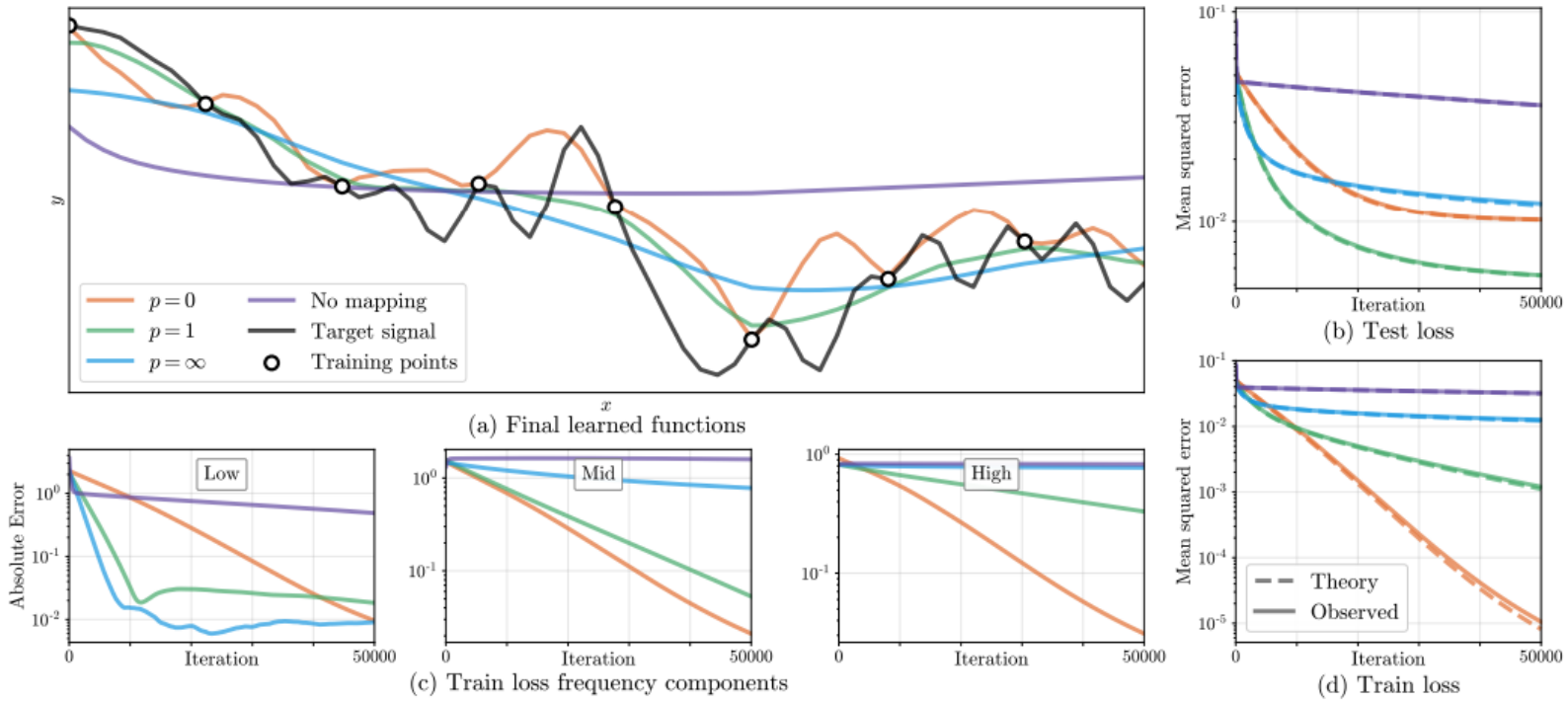
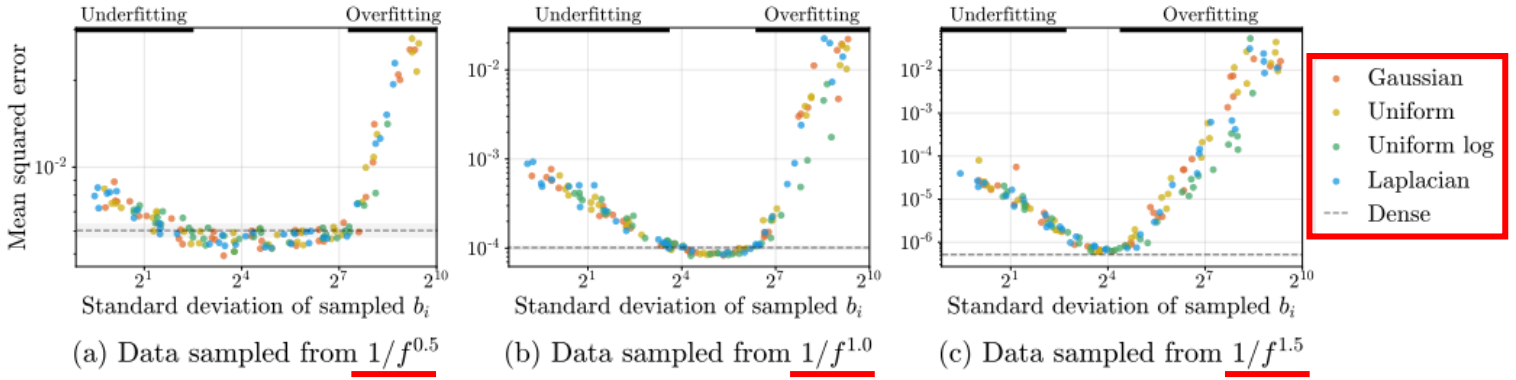


Figure 3: Combining a network with a Fourier feature mapping has dramatic effects on convergence and generalization.

Manipulate the Fourier Feature Mapping



(a) Data sampled from $1/f^{0.5}$ (b) Data sampled from $1/f^{1.0}$ (c) Data sampled from $1/f^{1.5}$

Figure 4: We find that a sparse random sampling of Fourier features can perform as well as a dense set of features and that the width of the distribution matters more than the shape. Here, we generate random 1D signals from $1/f^{\alpha}$ noise and report the test-set accuracy of different trained models that use a sparse set (16 out of 1024) of random Fourier features sampled from different distributions. Each subplot represents a different family of 1D signals. Each dot represents a trained network, where the color indicates which Fourier feature sampling distribution is used. We plot the test error of each model versus the empirical standard deviation of its sampled frequencies. The best models using sparsely sampled features are able to match the performance of a model trained with dense Fourier features (dashed lines with error bars). All sampling distributions trace out the same curve, exhibiting underfitting (slow convergence) when the standard deviation of sampled frequencies is too low and overfitting when it is too high. This implies that the precise shape of the distribution used to sample frequencies does not have a significant impact on performance.

References

- On the Spectral Bias of Neural Networks, Nasim Rahaman, Aristide Baratin et al., ICML 2019, https://arxiv.org/abs/1806.08734
- Neural Tangent Kernel: Convergence and Generalization in Neural Networks, Arthur Jacot et al., https://arxiv.org/abs/1806.07572
- Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains, Matthew Tancik, https://youtu.be/h0SXP6lJxak