STAT 535

12/8/22

Lecture 20

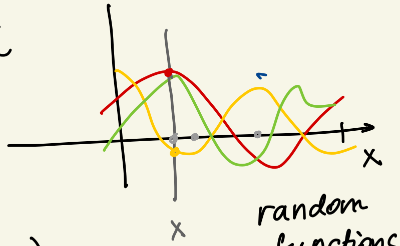
(... and last)

- Wide Neural networks as GP
- The NTK
- Deep NN $\left\langle \begin{array}{l} R \\ \text{Class} \end{array}\right.$

Gaussian Process

$\mathcal{X} \ni x$ distribution over functions on $\mathcal{X}$

Prior $GP(0, k)$



Posterior $GP(\mu_{x|D}, k_{|D})$

- $f(x) \sim N(0, k(x,x))$ for all x

  or known

- any set $x^{1:n}$, $f(\underbrace{x^{1:n}}_{X}) \sim N(0, G_X)$

define $\underline{k(x,x')} = cov(f(x), f(x')) \equiv \underline{E[f(x)f(x')]}$

Mercer kernel

Assume $k(x,x') = K(\|x - x'\|)$ e.g. Gaussian kernel

- $\underline{\underline{Pb}}$ know k

  observe $(x^{1:n}, y^{1:n} = f(x^{1:n})) = \mathcal{D}$

  Bayesian: Prior $f \sim GP(0, k)$

  Posterior $f \mid \mathcal{D} = ?$

$$\Sigma \equiv k_{xx,xx} = \begin{array}{cc} \begin{bmatrix} G & \cdot \\ k_{xX} & \end{bmatrix} & \begin{array}{l} \}\, X^{1:n} \\ x \end{array} \end{array}$$

$k(x,x)$

Same problem:

→ some $x \in \mathcal{X}$: wanted $f(x|\mathcal{D}) \sim N(\mu, \sigma^2)$
posterior

$\mu_{x|\mathcal{D}}$ $\quad$ $\sigma^2_{x|\mathcal{D}}$

Remark $\begin{bmatrix} y \\ y^{1:n} \end{bmatrix} \sim N(0_{n+1}; \Sigma)$ $\Big\}$ → want conditional $\quad y \,|\, y^{1:n}, X^{1:n} \sim N(\mu_{x|\mathcal{D}}, \sigma^2_{x|\mathcal{D}})$

jointly Gaussian

$$\mu_{x|\mathcal{D}} = \underbrace{[\,k(x,x^i) \cdots]}_{n}^{k(x,X)}\; \underbrace{G^{-1}}_{n\times n} \underbrace{[\,y^{1:n}\,]}_{n}$$

$$\sigma^2_{x|\mathcal{D}} = \underline{k(x,x)} - k(x,X)\, \underbrace{G^{-1}}_{n\times n}\, k(X,x)$$

Lecture VII – Wide multilayer networks and the Neural Tangent Kernel (NTK)

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

October, 2022

The Neural Tangent Kernel (NTK)

Wide networks and Gaussian Processes

The NTK is constant during training

Example – regression and $\mathcal{L}_{\mathrm{LS}}$

Wide and deep networks and classification

Notation

- Neural network predictor $f(x;\theta)$, where $x \in \mathbb{R}^d$
- For each layer $l = 1:L$ of dimension m_l, with $x^0 \equiv x$, and $z^L \equiv f(x)$

$$z^{l+1} = W^{l+1}x^l + b^{l+1} \qquad x^{l+1} = \phi(z^{l+1}) \tag{1}$$

  Here $x^{l,l+1}, z^{l+1}, b^{l+1}$ are column vectors W^{l+1} is a $m_{l+1} \times m_l$ matrix, $\phi()$ is the non-linearity/activation function.
- The weights

$$W^l_{ij} = \sigma_w w^l_{ij}/\sqrt{m_l}, \qquad b^l_j = \sigma_b \beta^l_j, \quad \text{Known as NTK parametrization} \tag{2}$$

- Parameter vector $\theta = \text{vector}\{w^{1:L}, \beta^{1:L}\} \in \mathbb{R}^p$ initialized i.i.d. $\sim N(0,1)$
- $\sigma_{w,b}$ are fixed hyper-parameters, $1/\sqrt{m_l}$ normalizes the expected norm of W^l columns
- Loss $\mathcal{L}(y,f)$

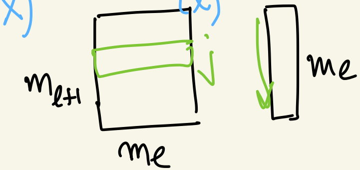
- We want to analize the behavior of this network $f()$ at initialization and during training, when $m_{1:L}$ very large
- Three approximations help analysis
  - (A1) continuous time training, called **gradient flow**
  - (A2) $m_{1:L} \to \infty$ in the wide limit, we can apply the Central Limit Theorem (CLT), and Gaussian Processes
  - (A3) parameters θ do not change much during training, i.e. $\theta_t - \theta_0$ is small

$x^0 = x$
input

$$x^\ell \longrightarrow \boxed{} \longrightarrow z^{\ell+1} \longrightarrow \boxed{\varphi} \longrightarrow x^{\ell+1} \dashrightarrow z^L = f(x)$$

$x^\ell \in \mathbb{R}^{m_\ell}$, $z^{\ell+1} \in \mathbb{R}^{m_{\ell+1}}$, φ, $x^{\ell+1} \in \mathbb{R}^{m_{\ell+1}}$, $z^L \in \mathbb{R}$ output

$$z^{\ell+1}(x) = W^{\ell+1} x^\ell(x) + b^{\ell+1}$$



$\ell = 1:L$

$z_j^{\ell+1}$

$j = 1:m_{\ell+1}$

Initialization

$$W_{jj'}^{\ell+1} \sim N\left(0, \frac{1}{m_\ell}\sigma_W^2\right)$$

$\sigma_W^2 \to 1$

$$b_j^{\ell+1} \sim N(0, \sigma_b^2)$$

σ_b^2: 1

1) $\underline{x}$ fixed, θ random

$\theta = (W^{1:L}, b^{1:L})$

$$z_j^{\ell+1}(x) \xrightarrow{CLT} z^{\ell+1} \sim N(0, \quad)$$

0) $x_{j'}^\ell(x)$ r.v $\in \mathbb{R}^{m_\ell}$ iid

$m_{1:L} \to \infty$ wide network

2) $z^\ell \sim N(0, \quad)$ multivariate gaussian

No proof 3) $z^\ell(x) \sim GP$ on $\mathcal{X}$

The NTK Neural Tangent Kernel

NTK

K^L for GPNN (at init for now)

layer ℓ : $z^\ell \sim GP(0, K^\ell)$

$f \equiv z^L \sim GP(0, K^L)$

training

The Gradient Flow

empirical Loss

$\mathcal{L}_\mathcal{D} = \left[\ \right]_i \in \mathbb{R}^n$

- Assume training by gradient descent on $\hat{\mathcal{L}} = \sum_i \mathcal{L}(y^i, f(x^i))$
- The gradient of $\hat{\mathcal{L}}$

$$\nabla_\theta \hat{\mathcal{L}} \;=\; \sum_i \underbrace{\frac{\partial \mathcal{L}}{\partial f}(y^i, f(x^i;\theta))}_{n}\nabla_\theta f(x^i,\theta) \;=\; \nabla_\theta f_\mathcal{D} \nabla_f \mathcal{L}_\mathcal{D} \qquad \in \mathbb{R}^p \tag{3}$$

where $\nabla_f \mathcal{L}_\mathcal{D} = [\frac{\partial \mathcal{L}}{\partial f}(y^i, f(x^i;\theta))]_{i=1:n} \in \mathbb{R}^n$, $\nabla_\theta f_\mathcal{D} = [\nabla_\theta f(x^i,\theta)]_{i=1:n} \in \mathbb{R}^{p\times n}$

- Assume (A1) gradient descent with infinitezimal time steps. In other words, the parameters evolve by an ordinary differential equation

$\partial_f \mathcal{L}_\mathcal{D}$

n

$p \ \boxed{\nabla_{\theta_j} f(x^i)}$ $\quad j \quad i$

$\frac{\partial \theta}{\partial t}$:

$$\dot\theta \;=\; -\eta \boxed{\nabla_\theta f_\mathcal{D}}\,\boxed{\nabla_f \mathcal{L}_\mathcal{D}} \qquad \in \mathbb{R}^p \tag{4}$$

$$\dot f(x) \;=\; \sum_{j=1}^{p} \frac{\partial f}{\partial \theta_j}\frac{\partial \theta_j}{\partial t} \;=\; (\nabla_\theta f)^T \dot\theta \qquad \in \mathbb{R} \tag{5}$$

$f_\mathcal{D} = \left[f(x^i)\right]_{i=1:n}$ - - ->

$$\dot f_\mathcal{D} \;=\; -\eta \underbrace{(\nabla_\theta f_\mathcal{D})^T \nabla_\theta f_\mathcal{D}}_{G} \nabla_f \mathcal{L}_\mathcal{D} \qquad \in \mathbb{R}^p \tag{6}$$

Gram matrix

- $G \equiv \nabla_\theta f_\mathcal{D}^T \nabla_\theta f_\mathcal{D} \equiv \kappa(X, X)$ is a Gram matrix!
- Therefore, we define the **Neural Tangent Kernel (NTK)** by

$$\kappa(x, x') \;=\; \nabla_\theta f(x;\theta)^T \nabla_\theta f(x';\theta) \tag{7}$$

tangent to f in θ space

$G_{ii'} = \left[\kappa(x^i, x^{i'})\right]_{i,i'=1:n}$

Gradient flow and NTK – summary

$$
\begin{aligned}
\dot{\theta} &= -\eta \nabla_\theta f_{\mathcal{D}} \nabla_f \mathcal{L}_{\mathcal{D}} && \in \mathbb{R}^p \\
\dot{f}_{\mathcal{D}} &= -\eta G \nabla_f \mathcal{L} && \in \mathbb{R}^p \\
\kappa(x, x') &= \nabla_\theta f(x)^T \nabla_\theta f(x')
\end{aligned}
$$

- f_X, $\nabla_\theta f_X$, G depend only on the inputs X, θ
- $\nabla_f \mathcal{L}_{\mathcal{D}}$ depends only on the correct outputs Y, and predicted outputs, i.e. on Y and θ
- This holds for any predictor! So what is special about neural networks?

Gradient flow and NTK – summary

$$
\begin{aligned}
\dot{\theta} &= -\eta \nabla_\theta f_{\mathcal{D}} \nabla_f \mathcal{L}_{\mathcal{D}} \qquad \in \mathbb{R}^p \\
\dot{f}_{\mathcal{D}} &= -\eta G \nabla_f \mathcal{L} \qquad \in \mathbb{R}^p \\
\kappa(x, x') &= \nabla_\theta f(x)^T \nabla_\theta f(x')
\end{aligned}
$$

- f_X, $\nabla_\theta f_X$, G depend only on the inputs X, θ
- $\nabla_f \mathcal{L}$ depends only on the correct outputs Y, and predicted outputs, i.e. on Y and θ
- This holds for any predictor! So what is special about neural networks?
- First, we will analyze κ for very wide neural networks with random parameters (e.g. at initialization) i $\equiv \kappa^L$ of GP
- Then, we will analyze what happens during training under assumption (A3)

iii

ii $m_{1:L} \to \infty \Rightarrow \kappa^{1:L}$ deterministic

Wide NN's Gaussian Process (GP)

- This is about f_0, a NN initialized with Gaussian independent parameters. For simplicity, we denote it as f.

- Assume $\theta^{1:L-1}$ fixed, only W^L, b^L random as in (**??**)
- Recall $f(x) = W^L x^{L-1}(x) + b^L$ for any x with $x^{L-1} \in \mathbb{R}^{m_L}$

- $f(x) =$ sum of m_{L-1} i.i.d. random variables, hence $f(x) \sim \mathit{Normal}$ by CLT, for m_{L-1} large
- Randomness is over weights W^L, b^L!!!
- We have $E[f(x)] = 0$ and

$$Cov(f(x), f(x')) = E[(W^L x^{L-1} + b^L)(W^L (x')^{L-1} + b^L)] = \frac{\sigma_w^2}{m_{L-1}} (x^{L-1})^T (x')^{L-1} + \sigma_b^2 \equiv \kappa^L(x \tag{8}$$

  where $x^{L-1}, (x')^{L-1} \in \mathbb{R}^{m_{L-1}}$ are the outputs of the $(L-1)$-th layer for inputs x, x'
- κ^L is a positive definite **kernel** Exercise Prove this.
- $f(x)$ is a random function of x
- The distribution of $f(x)$ defined as above, is called a **Gaussian Pocess**

- More generally, it can be shown [Jacot, Gabriel, Hongler, NeurIPS 2018] that, when all θ parameters are sampled as in (**??**), $f_0(x) \sim GP(0, \kappa^L)$

Q1 What is the kernel κ^L of this GP

Q2 This is all nice, but θ changes during training. What can we say about θ_t, f_t after training?

Q1: Idea. i

- From (8), for layer $l = 1 : L$ we have

$$\kappa^l(x, x') \;=\; E[z_j^l(x) z_j^l(x')] \;=\; \frac{\sigma_w^2}{m_{l-1}} (x^{l-1})^T (x')^{l-1} + \sigma_b^2 \tag{9}$$

with $x^{l-1} = \phi(z^{l-1})$. Note also that z_j^l are i.i.d. so it does not matter which j we choose.

- In particular, $\kappa^1(x, x') \;=\; \frac{\sigma_w^2}{m_1} x^T x' + \sigma_b^2$ is deterministic
- ...and κ^l is random for $l > 1$.
- However, when $m_l \to \infty$, $\frac{1}{m_{l-=}} (x^{l-1})^T (x')^{l-1} \to E[*]$
- More specifically, this expectation can be written as

$$E[*] \;=\; \int\int \phi(z)\phi(z') Normal\left(\left[\begin{array}{c} z \\ z' \end{array}\right]; 0, \kappa^{l-1}_{x,x'}\right) dz\, dz'. \tag{10}$$

In the above z, z' represent the $z^{l-1}(x), z^{l-1}(x')$ variables, sampled from the level l Normal distribution, which has covariance given by κ^{l-1}, namely

$$\kappa^{l-1}_{x,x'} \;=\; \left[\begin{array}{cc} \kappa^{l-1}(x, x) & \kappa^{l-1}(x, x') \\ \kappa^{l-1}(x', x) & \kappa^{l-1}(x', x') \end{array}\right]. \tag{11}$$

- Hence, the limit of $\kappa^l(x, x')$ when $m_{1:l} \to \infty$, is a **deterministic kernel** for all l. [Jacot, Gabriel, Hongler, NeurIPS 2018] derived this recursion (next page).

Q1: A recursive expression for the Neural Tangent Kernel

[Jacot, Gabriel, Hongler, NeurIPS 2018]

- L fixed, $m \to \infty$
- Simplified expression for $m_{0:L} = m$, $\sigma_w = \sigma_b = 1$
- Then the NTK $\kappa \equiv \kappa^L$ is defined recursively by layer

$$\kappa^1(x,x') = \Sigma^1(x,x'), \quad \Sigma^1(x,x') = \frac{1}{m}x^T x' + 1 \tag{12}$$

$$\kappa^{l+1}(x,x') = \kappa^l(x,x')\dot{\Sigma}^{l+1}(x,x') + \Sigma^{l+1}(x,x'), \tag{13}$$

$$\text{with} \quad \Sigma^{l+1}(x,x') = \mathrm{L}^{\phi}_{\Sigma^l(x,x')}, \tag{14}$$

$$\dot{\Sigma}^{l+1}(x,x') = \mathrm{L}^{\phi'}_{\Sigma^l(x,x')}, \tag{15}$$

$$\text{and} \quad \mathrm{L}^{\phi}_{\Sigma} = E[\phi(X)\phi(X')]\,\text{with}(X,X') \sim N(0, \begin{bmatrix} \Sigma(X,X) & \Sigma(X,X') \\ \Sigma(X,X') & \Sigma(X',X') \end{bmatrix} \tag{16}$$

- In other words, at level $l+1$, $X \equiv x^l, X' \equiv (x')^l$ are sampled from a GP with kernel Σ^l, and $\Sigma^{l+1}(x,x')$, $\dot{\Sigma}^{l+1}(x,x')$ represent their (scalar) covariance after passing through the non-linearities ϕ, ϕ' (where ϕ' is the derivative of ϕ)

Summary so far

- Now, we understand the random intialization of wide networks, with L layers.

$$f_0 \sim GP(0, \kappa^L) \tag{17}$$

where κ^L is a kernel that depends only on ϕ (and $\sigma^2_{b,w}$)

What next?

- Analysis of training by linearization iii
- Then, the NTK limit for $L \to \infty$ and its relevance for classification and regression iv

The Linearized Network $f^{\rm lin}$

$\| \theta_t - \theta_0 \|$ small after training

Notation: $\theta_{0,t}, f_{0,t}$ = parameters, predictor at times 0, t

- Here we use (A3), the assumption that the parameters θ change little during training. Extensive evidence supports this assumption.

- First order Taylor expansion of f_t around f_0

$$f_t^{\rm lin}(x) = f_0(x) + \nabla_\theta f_0(x)^T(\theta_t - \theta_0) \cong f^{\text{trained}} \quad (18)$$

non-linear in x, linear in θ

$$\nabla_\theta f_t^{\rm lin} = \nabla_\theta f_0 \quad (19)$$

$$\kappa(x, x') = \nabla_\theta f_0(x)^T \nabla_\theta f_0(x') \quad (20)$$

constant during training

$$G_0 \equiv \kappa_{X,X} = \text{NTK of random net} \quad (21)$$

$$\dot{\theta}_t = -\eta \nabla_\theta f_0(X)^T \nabla_f \mathcal{L}(Y, f_t^{\rm lin}(x)) \quad (22)$$

$$\dot{f}_t^{\rm lin}(x) = -\eta \underbrace{\kappa(x, X)^T G_0}_{\text{depends on } \theta_0} \underbrace{\nabla_f \mathcal{L}(Y, f_t^{\rm lin}(x))}_{\text{depends on } \theta_t} \quad (23)$$

NTK during training – empirical evidence

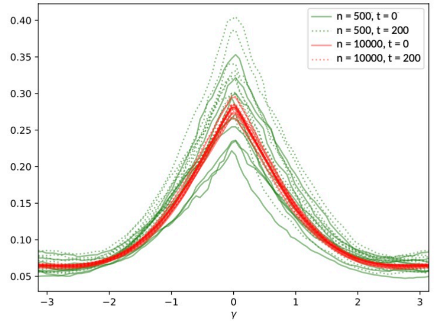


Figure 1: Convergence of the NTK to a fixed limit for two widths n and two times t.

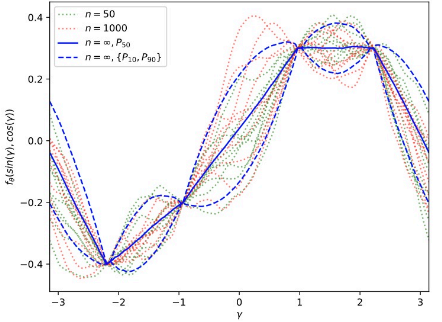


Figure 2: Networks function f_θ near convergence for two widths n and 10th, 50th and 90th percentiles of the asymptotic Gaussian distribution.

Linearized Network dynamics for $\mathcal{L}_{\mathrm{LS}}$

- For example, for $\mathcal{L}_{\mathrm{LS}}(y, f) = \frac{1}{2}(f-y)^2$, $\nabla_f \mathcal{L}_{\mathrm{LS}}(f, y) = f - y$. In this case, equations (**??**) are a linear system and have an analytic solution.

$$\theta_t - \theta_0 = -\nabla_\theta f_0(\mathrm{X})^T G_0^{-1} \left(I - e^{-\eta G_0 t}\right) (f_0(\mathrm{X}) - \mathrm{Y}) \tag{24}$$

$$f_t^{\mathrm{lin}}(\mathrm{X}) = \left(I - e^{-\eta G_0 t}\right) \mathrm{Y} + e^{-\eta G_0 t} f_0(\mathrm{X}) \tag{25}$$

$$f_t^{\mathrm{lin}}(x) = \underbrace{\kappa(x, \mathrm{X})^T G_0^{-1} \left(I - e^{-\eta G_0 t}\right) \mathrm{Y}}_{\mu(x)} + \underbrace{f_0(x) - \kappa(x, \mathrm{X})^T G_0^{-1} \left(I - e^{-\eta G_0 t}\right) f_0(x)}_{\gamma(x)} \tag{26}$$

Notes:

- if $G_0 \succ 0$ then $e^{-\eta G_0 t} \to 0$ for $t \to \infty$
- in discrete time $t = 0, 1, 3, \ldots$ replace e^{at} with $(1-a)^t$.
  Sketch of proof: $\ln(1-a)^t = t \ln(1-a) \approx t(-a)$ for a small; therefore $e^{-at} \approx (1-a)^t$.
- $f_t^{\mathrm{lin}}(x) = f_0(x) + \kappa(x, \mathrm{X})^T G_0^{-1} \left(I - e^{-\eta G_0 t}\right) (\mathrm{Y} - f_0(\mathrm{X}))$

Exercise Prove (**??**) from (**??**)

Wide and deep neural networks for classification – Basic quantities and assumptions

[Radhakrishnan, Belkin, Ulher, 2022]

- This paper studies the limits of wide neural networks $m_l \to \infty$ for all $l = 1 : L$ when the depth $L \to \infty$
- It is already known that for regression $L \to \infty$ is NOT OPTIMAL
- Since the NTK depends only of the activation function ϕ, the limit shall only depend on ϕ as well.
- In particular, the limit depends on ϕ only through the following

$$\boxed{\begin{array}{lll} A = E[\phi(Z)] & & \text{when } z \sim N(0,1) \\ A' = E[\phi'(Z)] & & \text{when } z \sim N(0,1) \\ & & \\ B = E[(\phi'(Z))^2] & & \text{when } z \sim N(0,1) \end{array}}$$

- Classifier $f(x) = \lim_{L\to\infty} \operatorname{sgn} \mathrm{Y} G^{-1} \kappa^L(\mathrm{X}, x)$ with $G = [\kappa^L(x^i, x^j)]_{i,j=1:n}$.

- Additional assumptions
  - Data $\mathrm{X} \subseteq \mathcal{S}_+^d$, vectors of norm 1 with all entries ≥ 0.
  - Simplifying assumptions on NTK parameters (e.g. $\sigma_w = \sigma_b = 1$)

"$\kappa^L \longrightarrow 0$" $\Leftarrow L \to \infty$

informal statement

Case $A \neq 0$: Networks implement majority vote

Theorem (Proposition 1 in [Radhakrishnan, Belkin, Ulher, 2022])

If there is a function $0 < c(L) < \infty$ *so that*

$$\lim_{L\to\infty} \frac{\kappa^L(x,x')}{c(L)} = c_1 > 0 \textit{ for any } x \neq x', \textit{ and } \lim_{L\to\infty} \frac{\kappa^L(x,x)}{c(L)} \neq c_1, \tag{27}$$

then

$$\lim_{L\to\infty} f(x) = \operatorname{sgn} \sum_{i=1}^{n} y^i \qquad \textit{MAJORITY CLASSIFIER} \tag{28}$$

- What ϕ's satisfy theorem? ReLU, all ϕ with $B \neq 1$.

Case $A = A' = 0$: Networks implement 1-nearest neighbor

Theorem (Theorem 3 in [Radhakrishnan, Belkin, Ulher, 2022])

Given x*, assume w.l.o.g. that* $x^T x^1 = \max_{i=1:n} x^T x^i$.

$$\lim_{L\to\infty} \frac{\kappa^L(x, x^i)}{\kappa^L(x, x^1)} = 0. \tag{29}$$

and

$$\lim_{L\to\infty} f(x) = \mathrm{sgn} y^1 \qquad \textit{1-nn} \tag{30}$$

Case $A = 0$, $A' \neq 0$: Networks implement singular kernel classifier

Theorem (Theorem 1 in [Radhakrishnan, Belkin, Ulher, 2022])

$$\lim_{L \to \infty} \frac{\kappa^L(x, x')}{(A')^{2L}(L+1)} = \frac{R(\|x - x'\|)}{\|x - x'\|^\alpha}, \tag{31}$$

with $\alpha = -4\frac{\log A'}{\log B'}$ *and* $R() \geq 0$*, bounded, and* $R(u) > \delta$ *around 0.*

- if $\alpha > 0$, $\frac{R(\|x-x'\|)}{\|x-x'\|^\alpha}$ is **singular kernel**
- Computationally not a problem: if data $x^{1:n}$ distinct, G_0 is well defined
- If $x = x^i$, set $f(x) = y^i$.

Optimality of singular kernel classifier

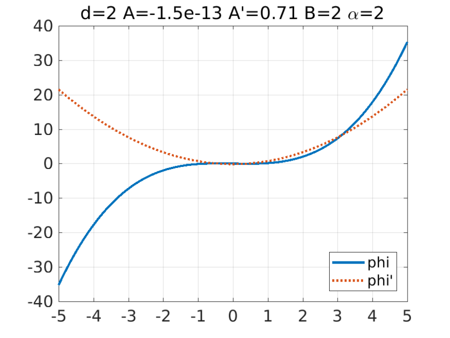
Theorem (Theorem 2 in [Radhakrishnan, Belkin, Ulher, 2022])

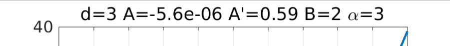
If $A = 0$, $A' \neq 0$ and $\boxed{\alpha = d}$ then $\lim_{L\to\infty} f(x)$ is Bayes-optimal.

- What activations ϕ satisfy this theorem?

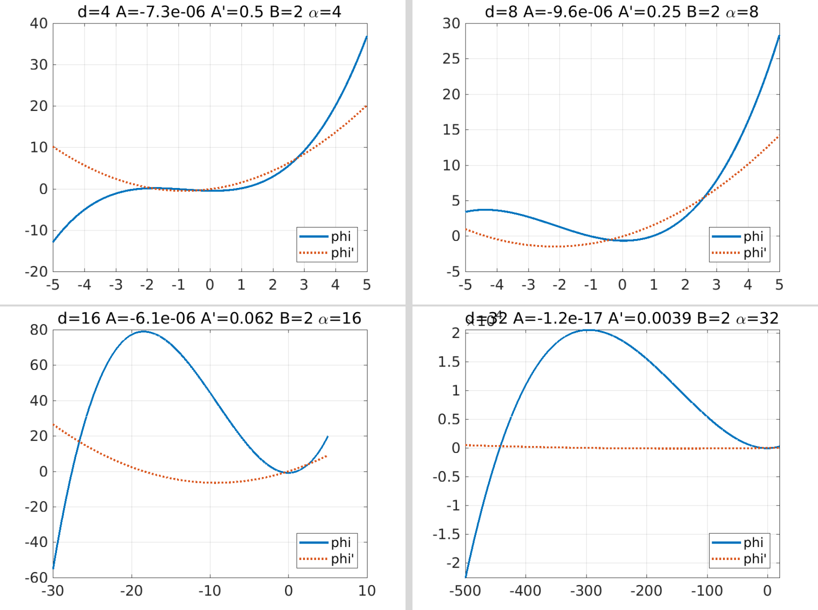
$$\phi^{\mathrm{opt}}(z) \;=\; \frac{1}{2^{d/4}}\frac{z^3-3z}{\sqrt{6}} + \sqrt{1-2^{1-d/2}}\,\frac{z^2-1}{\sqrt{2}} + \frac{1}{2^{d/4}}z \quad \text{for } d \geq 2$$

. (32)

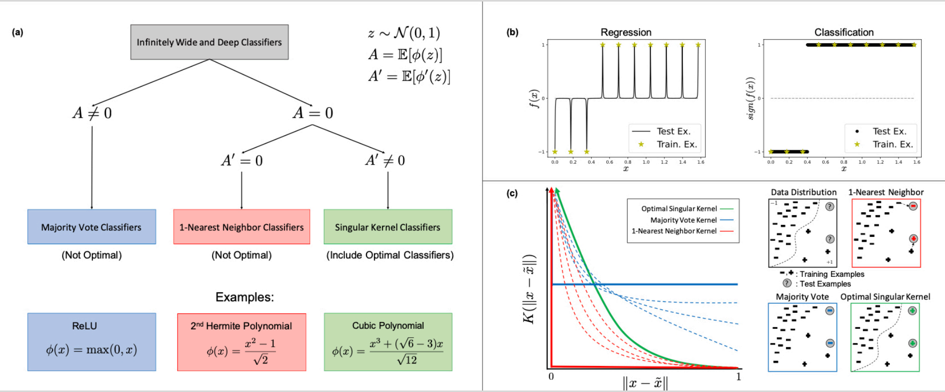




Optimal singular kernels for $d = 4, 8, 16, 32$



Summary



- b when $A \neq 0$, $\lim_{L\to\infty} \kappa^L(x, x') = 0$ for $x \neq x'$, and $f(x) = 0$ is vanishingly small (useless for regression), but $\mathrm{sgn} f(x)$ can be optimal for classification
- c Singular kernel $\alpha > d$,$\alpha < d$, majority vote kernel, and 1-nn kernel

Limits of some activation functions

$\phi^{\rm opt}$	Bayes classifier
ReLU	majority vote
sigmoid $\frac{1}{1+e^{-z}} - \frac{1}{2}$	1-nearest neighbor

Back of Form; Q1

TAs ability to help you when you interacted with him

A=poor……E=excellen

Please share any feedback for the TA

Back of yellow form