TUESDAY 08/03: TA SESSION 3 (THEODOR MISIAKIEWICZ)

GOING BEYOND THE LINEAR REGIME

1) Tractability via overparametrization 2) Double descent 3) Benign overfitting

→ (A) Separation between linearized NNs vs NNs (self-induced regularization)

→ (B) Fixed features vs feature learning:
breaking the curse of dimensionality using feature learning

→ (C) An example: learning parities

→ (D) Math approaches beyond linear regime

---

(1) Separation between linearized NNs vs NNs

LINEAR REGIME: training regime where network can be approximated by a linear model during the whole training dynamics

$$\Rightarrow f(x,\theta) \longleftrightarrow f^{lin}(x,\theta) = f(x,\theta_0) + \langle \theta - \theta_0, \underline{\nabla_\theta f(x,\theta_0)} \rangle$$

$\underline{\theta^0 = \bar{\theta}^0} = \theta_0$

(1) $\dot{\theta}^t = -\nabla \hat{R}_n(f(x,\theta^t))$

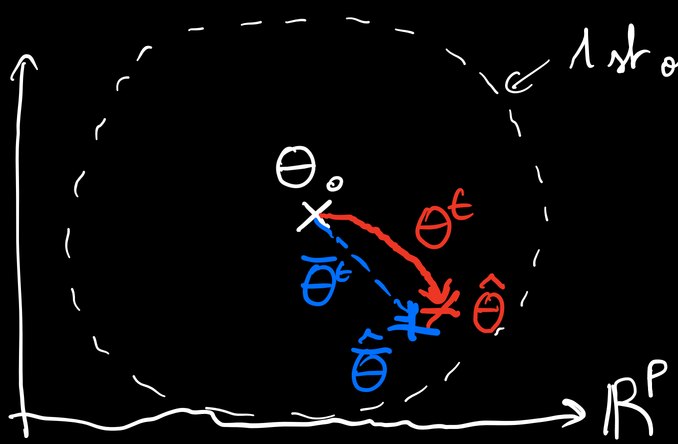
(2) $\dot{\bar{\theta}}^t = -\nabla \hat{R}_n(f^{lin}(x,\bar{\theta}^t))$

$\underline{\| \theta^t - \bar{\theta}^t \|_2 << 1}$

$\underline{\| \bar{\theta}^t - \bar{\theta}^0 \|_2 \text{ small}}$

$\Rightarrow \underline{f(x,\theta^t) \approx f^{lin}(x,\bar{\theta}^t)}$

$\Rightarrow$ in this regime: NNETs can effectively be replaced by linear models



1) Are NNETs in practice trained in the linear regime?

$\longrightarrow$ Sometimes, mostly not.

2) Does linear theory capture what can be achieved by NNETs?

$\longrightarrow$ No.

3) Do we have a better theory? $\rightarrow$ understand both optimization and generalization.

$\longrightarrow$ Not yet.

Linear regime: $\rightarrow$ explains why GD/SGD can find a global optima of a highly non-convex problem

Successfully illustrated: $\boxed{\text{TRACTABILITY VIA OVERPARAMETRIZATION}}$

$\rightarrow$ problem becomes more tractable as # of parameters $\nearrow$

$\Rightarrow$ since then: lots of work to show the limitation of linear regime theory to explain good generalization of NNETs

Most of the (theoretical) work: show in specific examples that NNETs outperform linearized NNETs.

This results are called "seperation results".

"Obvious" seperation in approximation power:

2-layer NNET: $\quad f_{NN}(x, a, W) = \sum_{i=1}^{N} a_i \, \sigma(\langle w_i, x \rangle)$

$a_i \in \mathbb{R}, \quad w_i \in \mathbb{R}^d$

2-layer linearized NNET: fix $W^0 = (w_1^0, \ldots, w_N^0)$

(RF) $\quad f_{RF}(x, a) = \sum_{i=1}^{N} a_i \, \sigma(\langle w_i^0, x \rangle)$ also apply full linearizat°

* $F_{NN}(B) = \{ f_{NN}(x, a, W) : \|a\|_2, \|W\|_F \leq B \}$

* $F_{RF}(W^0) = \{ f_{RF}(x, a) : a \in \mathbb{R}^N \}$

test error $n \to \infty$

Approximation error: $\boxed{R_{App}(f_*, F) = \inf_{f \in F} \| f_* - f \|_{L^2}^2}$

$\hookrightarrow$ best you can hope for any number of samples

Take: $x \sim \mathrm{Unif}(\mathbb{S}^{d-1})$

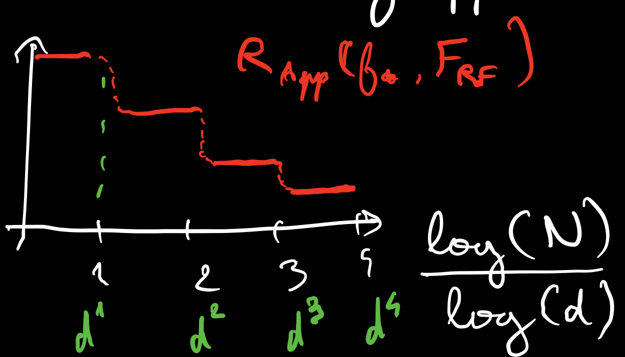
$w_i^0 \underset{iid}{\sim} \mathrm{Unif}(\mathbb{S}^{d-1})$ fixed

best degree-ℓ pol. approx.

Thm: [Misiakiewicz et al., 2019]

$$\boxed{\begin{array}{l}\text{For any } f_* \in L^2(\mathbb{S}^{d-1}). \text{ If } d^\ell \ll N \ll d^{\ell+1}, \text{ then} \\ R_{app}(f_*, F_{RF}(W^0)) = \|P_{>\ell} f_*\|_{L^2}^2 + o_d(1).\end{array}}$$

$\longrightarrow$ can only approximate degree ℓ polynomials



Staircase phenomena.

$\Longrightarrow$ Simple example: one neuron $f_*(x) = \sigma(\langle w_*, x \rangle)$

App. with RF: $R_{App}(f_*, F_{RF}) \simeq \|P_{>\ell}\sigma\|_{L^2}^2$ if $N \asymp d^\ell$

App. with NNETs: $R_{App}(f_*, F_{NN}) = 0$ for $N \geq 1$.

$\hookrightarrow$ simply take $a_1 = 1$, $w_1 = w_*$ + rest set to 0

$\Longrightarrow$ F_{NN} much richer class of fcts

Intuition: in high-dim, $\sup_{i \in [N]} |\langle w_i^0, w_* \rangle| \approx \frac{1}{\sqrt{d}}$

$\sigma(\langle w_*, \cdot \rangle)$ using $\sigma(\langle w_i, \cdot \rangle)$

→ no 'good' feature $\sigma(\langle w_i^0, x \rangle)$ to approximate $\sigma(\langle w_*, x \rangle)$

→ when 1st layer is not fixed, can select 'good' features (i.e. w_i high correlat° with w_*)

Kind of obvious and not interesting: the fact that you can approximate does not mean that you can efficiently find these good networks.

Want a separation between linearized NNETs and NNETs that can be "constructed" in practice, e.g., using GD.

Separation between linearized NNETs and gradient trained NNETs:

Inner-Prod kernel: 'infinite-width' linearized NNET

$$\underbrace{\frac{1}{N} \sum_{i=1}^{N} \sigma(\langle x, w_i^0 \rangle)\, \sigma(\langle z, w_i^0 \rangle)}_{} \xrightarrow[N \to \infty]{} \mathbb{E}_{w^0}[\sigma(\langle x, w^0 \rangle)\, \sigma(\langle z, w^0 \rangle)]$$

$$=: h(\langle x, z \rangle)$$

Test error of KRR (more details later about Kernel methods)

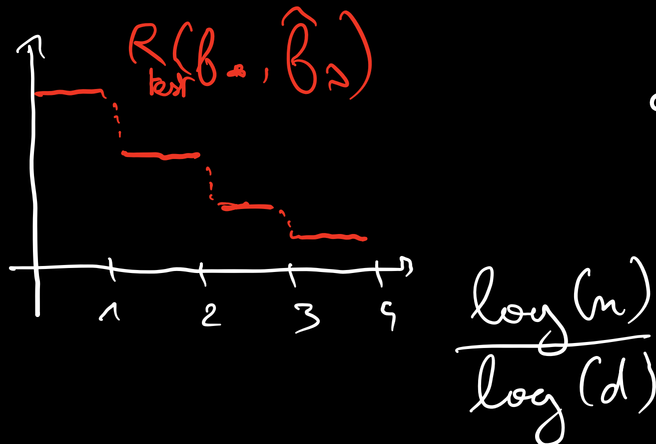
$$R_{test}(f_*, \hat{f}_\lambda) = \mathbb{E}_x\left[(f_*(x) - \hat{f}_\lambda(x))^2 \right]$$

Take: $x \sim \mathrm{Unif}(\mathbb{S}^{d-1})$ $\quad f_* \in L^2(\mathbb{S}^{d-1})$

If $d^\ell \ll n \ll d^{\ell+1}$ (n = # training samples)

Then [Misiakiewicz et al., 2019]

$$R_{test}(f_*, \hat{f}_\lambda) = \| P_{>\ell} f_* \|_{L^2}^2 + o_d(1)$$



Staircase descent

Again: $f_*(x) = \sigma(\langle w_*, x \rangle)$

$P_{>1}\sigma = 0$ $\quad n \gg d$

then $R_{test}(f_*, \hat{f}_\lambda) \simeq \| P_{>\ell}\sigma \|_{L^2}^2$ $\quad n \propto d^\ell$.

Take $\hat{f}_1(x) = \sigma(\langle w_1, x \rangle)$ $\quad$ 1 hidden unit

$\longrightarrow$ train w_1 using GD then if $n \gtrsim d \log d$

$$R_{test}(f_*, \hat{f}_1) \simeq 0$$ [Montanari et al., 2017]

$\Longrightarrow$ GD in underparametrized regime ($N = 1 \ll n$)

↳ very simple case but already very technical proof.

⟹ In general, studying NNETs trained by GD is currently out of reach except in the linear regime.

⟹ Can we understand the benefit of training more abstractly?

(B) Fixed features vs feature learning

→ $f_{RF}(x, a) = \sum_{i=1}^{N} a_i \, \sigma(\langle w_i^0, x \rangle)$

→ Kernel $\quad \underline{\frac{1}{N} \sum_{i=1}^{N} \sigma(\langle w_i^0, x \rangle)\sigma(\langle w_i^0, z \rangle) = \underline{H_0(x,z)}}$

$$f_{NN}(x, \underline{\theta}^t) = \sum_{i=1}^{N} a_i^t \, \sigma(\langle w_i^t, x \rangle)$$

→ "Kernel": $\quad \frac{1}{N} \sum_{i=1}^{N} \underline{\sigma(\langle w_i^t, x \rangle)\sigma(\langle w_i^t, z \rangle)} = \underline{H_t(x,z)}$

⟹ GD is a way to "training" a good kernel

$H_0 \downarrow H_t$ i.e., learning "good" features adapted to the data.

<u>e.g.</u>: $f_* = \sigma(\langle w_*, \cdot \rangle) \longrightarrow \underline{H_*(x,z) = \sigma(\langle w_*, x \rangle)\sigma(\langle w_*, z \rangle)}$

- <u>Linear regime</u>: "kernel regime", "lazy regime"
- <u>outside linear regime</u>: "feature learning regime"
  "rich regimes"

Vastly different behavior between fixed feature (fixed kernel) and methods that allow 'feature learning'

$\Rightarrow$ they are "adaptive" and can vastly outperform fixed feature models.

<u>"Breaking the curse of dimensionality using convex NNets"</u>

– Francis BACH (2017)

<u>Background on KRR / RKHS:</u>

$\{(y_i, x_i)\}_{i \leq n}$ $\quad y_i \in \mathbb{R}, \quad x_i \in \mathbb{R}^d = \mathcal{X}$

$(\mathcal{X}, \mathbb{P})$: proba space of the data covariates

(Ω, μ) : proba space of the features weights weights

<u>Featurization map</u>: $\sigma : \mathcal{X} \times \Omega \longrightarrow \mathbb{R}$

$$(x, \omega) \longmapsto \sigma(\langle x, \omega \rangle)$$

$(\sigma \in L^2(\mathcal{X} \times \Omega))$

<u>Model</u>: $\underline{f(x,a) = \int_\Omega a(\omega)\, \sigma(\langle x,\omega\rangle)\, \mu(d\omega)}$

$\rightarrow$ infinitely-wide 2 layers NNet with $\underline{a : \Omega \to \mathbb{R}}$

<u>e.g.</u> $a(\omega) = \sum_{i=1}^{N} a_i \, \delta_{\omega=\omega_i}$ (dirac) $\leftarrow$

then $f(x,a) = \underline{\sum_{i=1}^{m} a_i\, \sigma(\langle x, \omega_i\rangle)}$

<u>Define norm</u>: $\boxed{\|f(\cdot,a)\|_{F_2} = \left(\int_\Omega |a(\omega)|^2 \mu(d\omega)\right)^{\frac{1}{2}} = \|a\|_{L^2}}$ $\boxed{F_2\text{- norm}}$

$F_2 = \{ f(\cdot,a) \text{ such that } \|f(\cdot,a)\|_{F_2} < \infty \}$

$\rightarrow$ Reproducing Kernel Hilbert Space (RKHS) $\leftarrow$

with kernel: $\underline{K(x,z) = \int_\Omega \sigma(\langle x,\omega\rangle)\, \sigma(\langle z,\omega\rangle)\, \mu(d\omega)}$

$\|f\|_H = \|a\|_{L^2}$

<u>Kernel Ridge Regression</u>:

$$\underline{\hat{a}} = \underset{\substack{a : \Omega \to \mathbb{R} \\ (a \in L^2(\Omega))}}{\operatorname{argmin}} \left\{ \underline{\sum_{i=1}^{m} (y_i - f(x_i,a))^2 + \lambda \|a\|_{L^2}^2} \right\}$$

$\rightarrow$ convex problem in $a \in L^2(\Omega) \Rightarrow$ but on ∞ te dimensional space

$\rightarrow$ <u>Tractable</u>: celebrated <u>representer theorem</u>

usually not tractable

$\Rightarrow$ the solution $\hat{a} \in \underbrace{\mathrm{Span}\{\sigma(\langle x_i, \cdot\rangle) : i \le n\}}_{\text{n-dim linear subspace}}$ $\in L^2(\Omega)$

<u>proof:</u> $a \in L^2(\Omega)$, consider subspace $V = \mathrm{span}\{\sigma(\langle x_i, \cdot\rangle) : i \le n\}$
$\subset L^2(\Omega)$

let $a = \underset{\in V}{a_V} + \underset{\in V^\perp}{a_\perp}$ space orthogonal to V

We have $f(x_i, a) = \int \sigma(\langle x_i, \omega\rangle)\, a(\omega)\, \mu(d\omega)$

$$= \langle \sigma(x_i, \cdot), a\rangle_{L^2(\mu)} = \langle \sigma(x_i, \cdot), a_V\rangle_{L^2}$$

and $\|a\|_{L^2}^2 = \|a_V\|_{L^2}^2 + \|a_\perp\|_{L^2}^2$

$$\Rightarrow \hat{a} = \underset{a = a_V + a_\perp}{\mathrm{argmin}} \Big\{ \underbrace{\sum_{i=1}^n (y_i - \langle \sigma(x_i, \cdot), a_V\rangle)^2}_{a_V} + \underbrace{\lambda \|a_V\|_2^2}_{a_V} + \underbrace{\lambda \|a_\perp\|_2^2}_{a_\perp} \Big\}$$

$\Rightarrow \hat{a}_\perp = 0$ hence $\hat{a} \in \mathrm{Span}\{\sigma(\langle x_i, \cdot\rangle) : i \le n\}$ $\square$

$\hat{a} = \hat{a}_V$

<u>Closed form solution</u>: $\left(\hat{a} = \sum_{i=1}^n c_i\, \sigma(\langle x_i, \cdot\rangle) \right.$ $\in V$

$$\boxed{f(x, \hat{a}) = \sum_{i=1}^n c_i\, K(x, x_i)}$$

$\left(K(x, x_i) = \int \sigma(\langle x, \omega\rangle)\, \sigma(\langle x_i, \omega\rangle)\, \mu(d\omega) \right)$

Denote $K_n = (K(x_i, x_j))_{i,j \le n} \in \mathbb{R}^{n \times n}$

$y = (y_1, \dots, y_n) \in \mathbb{R}^n$

$$\hat{c} = \underset{c \in \mathbb{R}^n}{\mathrm{argmin}} \left\{ \|y - K_n c\|_2^2 + \lambda c^T K c \right\} \Big)$$

$$\boxed{\hat{c} = (K_n + \lambda\, \mathrm{Id})^{-1} y} \quad \in \ell_2$$

What is the performance of KRR?

$\longrightarrow$ consider $\mathcal{G} = \{ f_* \text{ L-lipschitz} \}$

$$\sup_{f_* \in \mathcal{G}} R_{test}(f_*, \hat{f}_\lambda) \asymp n^{-\frac{1}{d}}$$

$\longrightarrow$ for the 'worst case', to get error $\leq \varepsilon$ we need $n \geq \left(\frac{1}{\varepsilon}\right)^d$

$\boxed{\text{CURSE OF DIMENSIONALITY}}$

KRR is adaptive to smoothness of the function

$\hookrightarrow$ smoother fcts will be easier to fit

e.g.: 1) previous theorem: to fit degree-ℓ polynomial

$\longrightarrow$ need $n \geq d^\ell$ $\leftarrow$ not here $\lambda \in [0, C]$

$\lambda = 0$ $\lambda \nearrow$ test error $\nearrow$ $\lambda \geq C$.

2) $\mathcal{G}_S = \{ f_* \text{ with } S \text{ first derivatives bounded} \}$ $\leftarrow$ Hölder space

good kernel $\rightarrow$ $$\sup_{f_* \in \mathcal{G}_S} R_{test}(f_*, \hat{f}_\lambda) \asymp n^{-\frac{S}{S+d}} \swarrow \lambda_*$$

$\Longrightarrow$ still need smoothness S to grow with d (curse of dim.)

$S \propto d$

Can we hope to do better?

→ No: these classes of fcts are too big ('plague by the curse of dim')

Need to restrict to a smaller class of fct

Interesting class of fcts: $\underset{f_*(x)}{=} g(Ux)$ $\quad U \in \mathbb{R}^{s\times d} \quad s \ll d$

→ fcts that only depend on x on a low-dimensional projection

Why? $\quad f(x, a) = \int \sigma(\langle x, \omega\rangle)\, a(\omega)\, \mu(d\omega)$

→ put all the weights $a(\omega)$ on $\omega \in \mathrm{Im}(U^\top)$

→ problem effectively s-dimensional $\quad d \to s$

↳ hope to get $O(n^{-\frac{1}{s}})$ rate.

However: $\boxed{\text{Kernel methods are not adaptive to fcts that depend only on a low-dimensional projection of data}}$

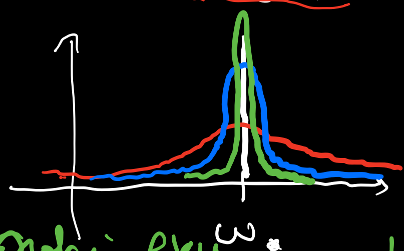
Recall: if $n \propto d^\ell$, $\quad R_{test}(f_*, \hat{f}) = \|P_{>\ell} f_*\|_{L^2}^2 + o_d(1)$

↳ no matter the structure on f_* (e.g. $= \sigma(\langle \omega_*, x\rangle)$)

$= \int a(\omega)\,\sigma(\langle \omega, x\rangle)\,\mu(d\omega)$

Intuition: in order to have $f(x, a) \to \sigma(\langle \omega_*, x\rangle)$

we need $a(\omega) \longrightarrow \delta_{\omega=\omega_*}$ $\quad(\sigma \notin \mathcal{F}_2)$

a density  we must have

$\| \delta_{\omega=\omega_*} \|_{L^2} = \infty$ $\qquad \|a\|_{L^2} \to \infty$

$\sigma(\langle \omega_*, \cdot\rangle)$ not in RKHS, σ not in F_2

$\hookrightarrow$ HOWEVER: $\int_\Omega |a(\omega)|\,\mu(d\omega) =: \|a\|_{L^1}$

remains bounded

$\hat{a}_k \to \delta_{\omega_0, \omega_*}$

$\|\hat{a}_k\|_{L^1} \leq 1.$

$\Rightarrow$ so .... why not replacing $\|a\|_{L^2}$ by $\|a\|_{L^1}$

i.e.

$\to$ $(F_2 - P)$ $\quad \hat{a} = \underset{a:\Omega\to\mathbb{R}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} (y_i - f(x_i, a))^2 + \lambda \|a\|_{L^2}^2 \right\}$

(Kernel method)

$\to$ $(F_1 - P)$ $\quad \hat{a} = \underset{a:\Omega\to\mathbb{R}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} (y_i - f(x_i, a))^2 + \lambda \|a\|_{L^1} \right\}$

$\to$ still convex $\qquad$ CONVEX NNETs

(might not be tractable) $\qquad f(x, \hat{a}_{F_1})$

$\to$ no representer thm

If $f_* = g(Ux)$, $\qquad R(f_*, \hat{f}_{F_1}) \leq n^{-\frac{1}{s}}$

$U \in \mathbb{R}^{s\times d}$

$\Rightarrow$ $\boxed{\text{Convex NNets break the curse of dimensionality on fcts that only depend on a low-dim projection of the data}}$

$\Rightarrow$ adaptive to latent linear structure (U is unknown)

However F_1 is not tractable (hard problem)

$\longrightarrow$ can think about GD as approximately solving F_1

$\Rightarrow$ in general do not expect GD to solve F_1 - problems (not the right implicit bias)

$\hookrightarrow$ However, one situation where GD was proven to solve approximately F_1 problem

"Implicit bias of GD for wide 2-layers NNets"
— Chizat and Bach, 2020.

Linear regime $\longrightarrow$ F_2: problem
- $\nearrow$ curse of dim
- $\rightarrow$ adaptive to smoothness
- $\searrow$ not adaptive to low-dim projectᵒ fcts

Sometimes

Non-linear dynamics $\longrightarrow$ F_1: problem
- $\nearrow$ adaptive to smoothness
- $\searrow$ adaptive to low-dim projectᵒ

$\Rightarrow$ break curse of dim. on these fct classes.

(C) An example: learning parities

So far, we saw

- Limitation of kernel methods/linear models
- Feature learning necessary to break the curse of dimensionality
- One "classical regime" example: GD fitting a single neuron with another neuron.

underparametrized

More realistic example where we can study feature learning with GD

$$x \sim \mathrm{Unif}(\{\pm 1\}^d)$$

learning class of k-parity fcts

$$\mathcal{C}_k = \{ f_A(x) = \prod_{i \in A} x_i \; : \; \forall A \subseteq [d], |A| = k \}$$

parity of subset A.

Hardness result of learning parity fcts with kernel methods

Prop: [Allen-Zhu et al., 2020] If for any $f_A \in \mathcal{C}_k$

$$R_{test}(f_*, \hat{f}_\lambda) \leq \frac{1}{9}$$

then we must have $n \geq \frac{3}{4} \binom{d}{k} \quad (\propto d^k)$

Rmk: 1) $f_A(x)$ is a degree-k polynomial, already implied by previous result if kernel is an inner-product kernel then need $n \gtrsim d^k$ samples

$\longrightarrow$ here very elementary proof for any kernel

2) $f_A(x)$ only depend on a low-dimensional projection of dimension k

$\longrightarrow$ expect F_1 problem to be able to efficiently learn $\mathcal{C}_k$

Proof: [If time, probably not: very nice proof using only elementary algebra]

"Learning parities with neural networks"
[Amit Daniely and Eran Malach, 2020]

With slightly different distribution

+ classification setting: $\ell(\hat{y}, y) = \max(1 - y\hat{y}, 0)$

+ 2 layers NNets with ReLu activations

Thm: for any linear model $\hat{f}(x) = \langle \psi(x), \hat{a} \rangle$ with $\psi(x) \in \mathbb{R}^N$ and $\|\hat{a}\|_2 \leq B$, then there exists $f_A \in \mathcal{C}_k$ such that

$$R_{test}(f_A, \hat{f}) \geq \frac{1}{2} - \frac{\sqrt{N}\, B}{2^k \sqrt{2}}$$

Thm: (informal) GD with some initialization and learning steps on population loss, for T steps with high probability, for any $f_A \in \mathcal{C}_k$

$$R_{test}(f_A, \hat{f}^{(T)}) \lesssim \frac{k^8}{\sqrt{N}} + \frac{Nk}{\sqrt{d}} + \frac{k^2\sqrt{N}}{T}$$

Rmk: 1) $k \propto d^{\frac{1}{64}}$, $N \propto k^{16} \propto d^{\frac{1}{4}}$, $T \propto d^{\frac{3}{4}}$

Then $\exists\, f_A \in \mathcal{C}_k$ such that

$$R_{test}(f_A, \hat{f}^{(lin)}) \geq \frac{1}{4}$$

$$R_{test}(f_A, \hat{f}^{(GD)}) \simeq 0.$$

2) Still unsatisfactory: here $n = \infty$ (or very large) + artificial GD learning steps schedule

Proof idea: Initialization

(0) $f(x, \Theta^{(0)}) = \sum_{j=1}^{N} a_j^{(0)} \sigma(\langle w_j^{(0)}, x \rangle)$

1 large GD step

(1) $f(x, \Theta^{(1)}) = \sum_{j=1}^{N} a_j^{(1)} \sigma(\langle w_j^{(1)}, x \rangle)$

* learn good weights $w_j^{(1)}$ with large correlation with A
* show that if we fix $w_j^{(1)}$ and only train $a_j^{(1)}$, can fit f_A

(2→T) Following learning steps sufficiently small such that we are in the linear regime

Summary:

GD steps

(0) Initialize at $a_i^{(0)}$, $w_i^{(0)}$

(1) One gradient step learn good $w_i^{(1)}$

(2-T) Fit second layer $a_i^{(T)}$ while $w_i^{(1)}$ almost fixed

$\square$

D) Going beyond the Linear regime: mathematical approaches

1) Higher-order Taylor expansion around initialization (Dan Roberts et al. 2021)

$GD \to K_t \to K_t \to K_t^{(2)}$

2) Can see GD dynamics as kernel dynamics with time varying kernel $K_t(x,z)$

$\longrightarrow$ can write ODE for K_t

$\longrightarrow$ hierarchy of ODEs with higher order kernels $K_t^{(k)}$

$\longrightarrow$ can truncate at some level $K_t^{(k)} = K_0^{(k)}$ is fixed

$K_t \downarrow K_t^{(2)} \downarrow \dots$

completely non linear

3) Mean-Field dynamics: $\theta^{(t)}$ weights after t SGD steps

$$f_N(x, \theta^{(t)}) = \frac{1}{N} \sum_{i=1}^{N} a_i^{(t)} \sigma(\langle \omega_i^{(t)}, x \rangle) = \int a(\omega)\, \sigma(\langle \omega, x \rangle)\, \hat{\rho}_t(d\omega)$$

(where $\hat{\rho}_t(d\omega) = \frac{1}{N} \sum_{i=1}^{N} \delta_{\omega = \omega_i^{(t)}}$)

$$\longrightarrow f(x, \rho_t) = \int a(\omega)\, \sigma(\langle \omega, x \rangle)\, \rho_t(d\omega)$$

PDE on ρ_t: evolution in the space of measures

→ send paper: on Multi-layer MF.

→ Marco Mondelli

generalizat° error of

$f(x, \theta^t) \longrightarrow \theta^t$

NTK: FC multilayer NN

weights $\sim N(0, Id)$

→ NTK $h(\langle x, y \rangle)$ inner prod

1) learn only low d° poly on the sphere

2) F_2-P → not adaptive

→ NTK: KRR with NT kernel

① $x \in \mathbb{R}^d$ $x = Uz + U_1 z'$

$x = Uz$ $U \in \mathbb{R}^{d \times D}$ low variance

$$h(\langle x_1, x_2 \rangle) = h(\langle z_1, U^T U z_2 \rangle)$$

$$= h(\langle z_1, z_2 \rangle)$$

$\longrightarrow$ dim D