# Admrnistative Info

Lectures will be recorded and posted on slack

Assessment: Two problem sets (20% + 20%)

Final project (40%)

Independent research and/or literature survey related to course material

Later in the semester I will want to meet/discuss with you

Participation (20%)

We will share one materials, lectures with Harvard "sister" class – more emphasis on experiments

Important Note: I do not (currently) work on DL, so this class will be my attempt to organize (and simplify) what I think is important, but grounded in classic learning theory

Let's pick up where we left off last time:

<u>Proposition</u>: For $\|x\| \le 1$ we have

$$f(x) - f(0) = -\int \int_{0}^{\|w\|} \mathbb{1}_{w^T x \ge b} \frac{\sin(2\pi b + 2\pi \theta(w))}{\|w\|} \|\widehat{\nabla f}(w)\| \, db \, dw$$

$$+ \int \int_{-\|w\|}^{0} \mathbb{1}_{w^T x \le b} \frac{\sin(2\pi b + 2\pi \theta(w))}{\|w\|} \|\widehat{\nabla f}(w)\| \, db \, dw$$

where $\hat{f}(w) = |\hat{f}(w)| e^{2\pi i \theta(w)}$, i.e. $\theta(w)$ is the angle

First, why is this useful?

It is an infinite width, depth two rep.

$$f(x) - f(0) = \int\int \mathbb{1}_{w^T x \ge b} \mu_1(w,b) \, db \, dw + \dots$$

So let's estimate $\|\mu_1\|_1 + \|\mu_2\|_1$:

$$\leq \int\int_{0}^{\|w\|} \frac{\|\widehat{\nabla f}(w)\|}{\|w\|} \, db \, dw + \int\int_{-\|w\|}^{0} \frac{\|\widehat{\nabla f}(w)\|}{\|w\|} \, db \, dw$$

$$\leq 2 \, \|\widehat{\nabla f}(w)\|_1$$

Thus we have:

Barron's Representation + Maurey's Lemma =

Barron's Theorem

All that remains is to prove the proposition:

<u>Proof:</u> Since $f$ is real-valued, we have

$$f(x) = \mathrm{Re}\left\{ \int e^{2\pi i w^T x} \hat{f}(w) \, dw \right\}$$

$$= \int \mathrm{Re}\left\{ e^{2\pi i w^T x + 2\pi \theta(w)} \right\} |\hat{f}(w)| \, dw$$

$$= \int \cos\left(2\pi w^T x + 2\pi\,\theta(w)\right) |\hat{f}(w)|\, dw$$

This is called the polar decomposition

Next we write

$$f(x) - f(0) =$$

$$\int \left(\cos\left(2\pi w^T x + 2\pi\theta(w)\right) - \cos\left(2\pi\,\theta(w)\right)\right) |\hat{f}(w)|\, dw$$

$$\overset{(*)}{=} \int \frac{\cos\left(2\pi w^T x + 2\pi\,\theta(w)\right) - \cos\left(2\pi\,\theta(w)\right)}{2\pi \|w\|} \underbrace{\|\widehat{\nabla f}(w)\|}_{2\pi \|w\|\, |\hat{f}(w)|} dw$$

$$\cos\left(2\pi w^T x + 2\pi\,\theta(w)\right) - \cos\left(2\pi\,\theta(w)\right) =$$

$$\int_0^{w^T x} -2\pi \sin\left(2\pi b + 2\pi\,\theta(w)\right) db =$$

$$-2\pi \int_0^{\|w\|} \underline{\mathbb{1}}_{w^T x \geq b} \sin\left(2\pi b + 2\pi\,\theta(w)\right) db + \ldots$$

Plugging this expression back into $(*)$ completes the proof ⊠

## How rich is the class of Barron functions?

i.e. functions with $C_f = \mathrm{poly}(d)$, so that we avoid the curse of dimensionality

Important Example: Gaussians

Claim let $f(x) = e^{-\pi\|x\|^2}$, then $C_f \leq O(\sqrt{d})$

Proof: (by computation): First, the fourier transform of a Gaussian is a Gaussian

Explicitly, $\hat{f}(\omega) = e^{-\pi\|\omega\|^2}$

Note: The constants depend on your conventions for the fourier transform / inv.

I've chosen convenient constants here

In particular, both $f$ and $\hat{f}$ are Gaussians with variance $\frac{1}{2\pi}$

Now we can compute

$$C_f = 2\pi \int \|w\| \, |\hat{f}(w)| \, dw$$

$$\overset{\text{Jensen}}{\leq} 2\pi \left( \int \|w\|^2 \, |\hat{f}(w)| \, dw \right)^{\frac{1}{2}}$$

$$= 2\pi \left( d \times \text{variance} \right)^{\frac{1}{2}} = O(\sqrt{d})$$

▨

Taking a step back, recall the idea of a KDE was to

① take samples from a distribution

② smooth the empirical distribution by convolving by a bump

But even for simple distributions (like a Gaussian) this suffers from the curse of dimensionality

**But Barron's Theorem doesn't!**

Comment: $C_f$ as a complexity measure also behaves nicely under addition, dilate, etc

## A View from Optimization

How can we construct a succinct representation, as promised by Barron's theorem?

It turns out we can use the Frank-Wolfe method:

```
For $i = 1$ to $k$
    Suppose the current rep. is
        $g_{i-1} = \sum_{j=1}^{i-1} \alpha_j v_j$
```

Solve the problem:

$$\min_{\lambda, v_i} \quad \| \lambda g_{i-1} + (1-\lambda) v_i - x \|^2$$

$$\text{s.t. } \lambda \in [0,1], \quad v_i \in S$$

Lemma: Suppose $x \in \text{conv}(S)$. Then

$$\| g_i - x \|^2 \leq \frac{\max_{v \in S} \|v\|^2}{i}$$

Proof: suppose the bound holds for $g_{i-1}$. Now guess a (potentially suboptimal):

$$\hat{g}_i = \lambda g_{i-1} + (1-\lambda) v_i$$

where $v_i \sim \mu$, and $\mathbb{E}_\mu [v] = x$

Now we can compute

$$\| g_i - x \|^2 \leq \mathbb{E} [ \| \hat{g}_i - x \|^2 ]$$

$$= \mathbb{E}\left[\|\lambda(g_{i-1}-x) + (1-\lambda)(v_i-x)\|^2\right]$$

$$= \lambda^2\|g_{i-1}-x\|^2 + (1-\lambda)^2\mathbb{E}\left[\|v_i-x\|^2\right]$$

$$\le \lambda^2\|g_{i-1}x\|^2 + (1-\lambda)^2\,\mathbb{E}\left[\|v_i\|^2\right]$$

Now plugging in $\lambda = 1-\frac{1}{i}$ and our inductive bound, we get

$$\le \underbrace{\left(\frac{\lambda^2}{i-1} + (1-\lambda)^2\right)}_{=\frac{1}{i}} \max_{v\in S}\|v\|^2 \qquad \blacksquare$$

But there is a catch!

Even if you fix $\lambda$, finding the best $v_i$ is (?) computationally hard!

# Depth Separations

So far, we've identified interesting classes of functions that can be well-approximated by shallow (depth=2) networks

**Main Question: Does increasing the depth buy you expressivity?**

Ideally, we want an algorithmic answer:

> "I can solve problem A with SGD on a depth d network, but not on a depth d-1 network"

We'll settle for the more modest goal of finding explicit functions that can be that can be represented w/ depth L, but not L-1

These are called depth separations

Main Theorem [Telgarsky] There is a function $f$ that can be computed by a ReLU network with:

$0 \leq f(x) \leq 1$

$$\text{depth} = O(L^2)$$

$$\#\text{units} = O(L^2)$$

but for any function $g$ computed with:

$$\text{depth} \leq L$$

$$\#\text{units} \leq 2^L$$

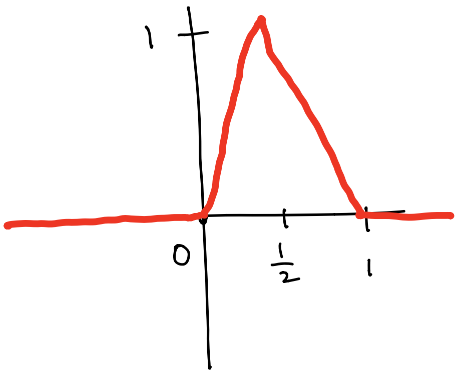
we must have: $$\int_{[0,1]} |f(x) - g(x)|\,dx \geq \frac{1}{32}$$

Recall, you can approximate any continuous function arbitrarily well by a depth two network, but: you can and do need exponentially many more units

Intuition: A ReLU network computes a function made up of many flat regions and:

# flat regions grow polynomially with width, but exponentially in depth

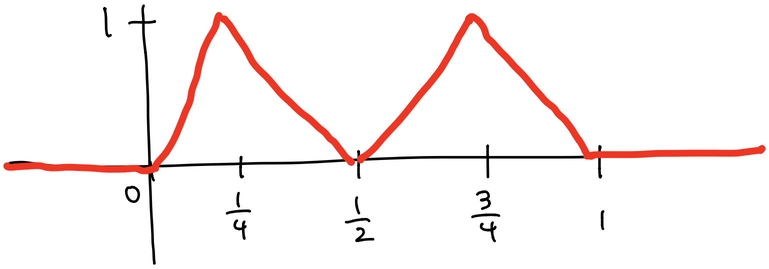
Let's see a simple construction where the # flat regions is exponential:



$$m(x) \triangleq \begin{cases} 2x & \text{if } 0 \le x \le \frac{1}{2} \\ 2-2x & \text{if } \frac{1}{2} < x \le 1 \\ 0 & \text{else} \end{cases}$$

**what happens if we iterate m(x)?**

i.e. $m(m(x)) \triangleq m^{[2]}(x)$



Similarly $m^{(n)}(x)$ has $2^n$ "teeth"

The key point is $m(x)$ can be computed by a simple depth two ReLU network:

<u>claim</u>: $m(x) = \sigma(2\sigma(x) - 4\sigma(x - \frac{1}{2}))$

Now let's introduce a notion of complexity:

<u>def</u>: we say a function $f: \mathbb{R} \to \mathbb{R}$ is a <u>$t$-sawtooth</u> if it is piecewise affine with $t$ pieces

It is easy to see that

① $m$ is a 4-sawtooth

② $m^{(2)}$ is a 6-sawtooth

... and more generally, $m^{(n)}$ is a $2^n+2$ - sawtooth

The key point is that simple operations do not grow the complexity too much:

Lemma: Suppose that $f$ and $g$ are $s$- and $t$-sawtooths respectively. Then

(1) $af+b$ is at most an $s$-sawtooth

(2) $f+g$ is at most an $s+t$-sawtooth

(3) $f\circ g$ is at most an $st$-sawtooth

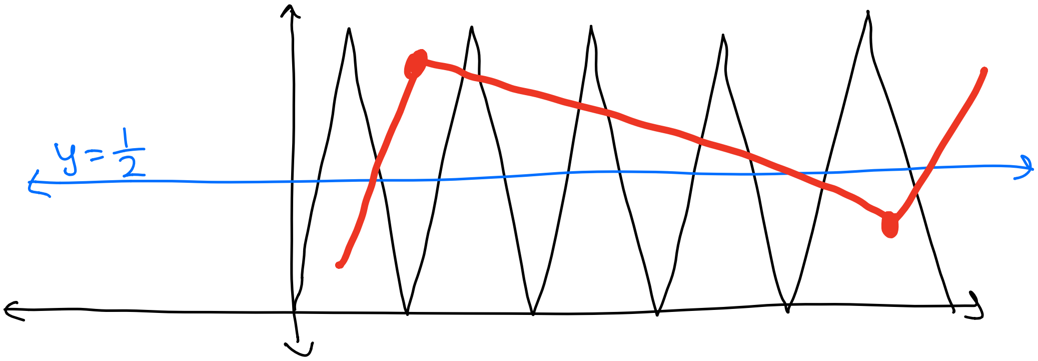
Corollary: If $g$ is computed by a depth $L$ ReLU network with at most $2^L$ hidden units then $g$ has sawtooth complexity at most

$$2^{O(L^2)}$$

Intuitively, this means we should not be able to approximate $m^{(O(L^2))}$

**But how do we lower bound its error?**

Proof by Picture: To be concrete, consider $m^{(d)}$ for $d = L^2+2$. Let's count the triangles that any purported approximator $g$ misses



There are $2^{L^2+2}-1$ triangles, each with area:

$$\frac{1}{2} \times \frac{1}{2} \times \frac{1}{2^{L^2+2}} = 2^{-L^2-4}$$

Now consider an affine piece of g. Other than its boundary, it misses half the triangles

Thus we have:

$$\int_{[0,1]} |f(x)-g(x)|\,dx \geq \left(\begin{matrix}\#\text{ missed}\\ \text{triangles}\end{matrix}\right)(\text{area})$$

$$\geq \frac{1}{2}\left(2^{L^2+2}-1-2\cdot 2^{L^2}\right)\left(2^{-L^2-4}\right)$$

$$\geq \frac{1}{32}$$ ▨

There are many other separations in the literature:

[Bengio, Delalleau]: Depth separations for sum-product networks — essentially, separating polynomials by degree

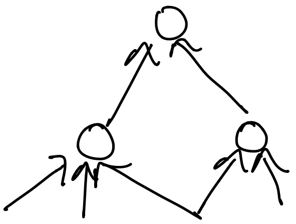
[Eldan, Shamir]: Separation of depth two vs. depth three via fourier analysis

[Lee, Ge, Ma, Risteski, Arora]: Higher depth analogue of Barron functions, show richness of class by proving $C_f$ is exponential in $d$

Digression to Circuit Complexity

For Boolean functions, **HUGE** gaps between what we can separate explicitly vs. non-explicitly

def: The class $TC^0$ is the set of circuits



that can be computed by unbounded

fan-in, constant depth circuits with threshold gates, i.e. $y = \text{sgn}(w^T x + b)$

From counting arguments we know:

$\exists$ functions that can be computed by polynomial sized depth $d$ circuits, but, but can only be trivially approx. by depth $d-1$ threshold circuits

This is "inexplicit" in the sense that we don't know what they look like

Best known explicit constructions are much worse (and if we could do better, we'd likely get better PRGs)

# Optimization

Thus far we have discussed:

1. How expressive are deep nets?

2. Does depth allow us to express functions more succinctly?

The second pillar of the course will be about:

3. How do we fit a deep net to data?

The main approach will be to cast it as a giant and unwieldy optimization problem:

Given data $(x_1, y_1) \ldots (x_N, y_N)$, want to solve:

$$\underset{W}{\arg\min} \sum_i \ell(\hat{y}_i, y_i)$$

where W are the parameters, e.g.

$$W = (w_1, b_1, \ldots W_L, b_L)$$

for the basic model

and $\hat{y}_i$'s are the predictions, e.g.

$$\hat{y}_i \triangleq f(x_i; w)$$

To keep it simple, we will consider the squared loss

$$R(f(x;w)) \triangleq \frac{\|\hat{y} - y\|^2}{2}$$

And we will sometimes call this <u>risk</u>

The trouble is this optimization problem doesn't fall into the class of problems where we have strong guarantees

Let's discuss the classic theory first

# Convex Optimization

<u>def:</u> we say that a function $f: \mathbb{R}^d \longrightarrow \mathbb{R}$ is <u>convex</u> if

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$$

for all $x, y \in \mathbb{R}^d$ and $0 \leq \lambda \leq 1$

In <u>unconstrained convex optimization</u>:

Given: convex, differentiable $f$

Output: A minimizer $x^* = \underset{x}{\operatorname{argmin}} f(x)$

sometimes can only get $\varepsilon$-approx. min

## what makes a convex function nice?

<u>def:</u> we say that $D \subseteq \mathbb{R}^d$ is <u>convex</u> if

$$\lambda x + (1-\lambda) y \in D$$

for all $x, y \in D$ and $0 \leq \lambda \leq 1$

The important point is the sublevel sets of a convex function, i.e.

$$D = \{x \mid f(x) \leq C\}$$

are all convex

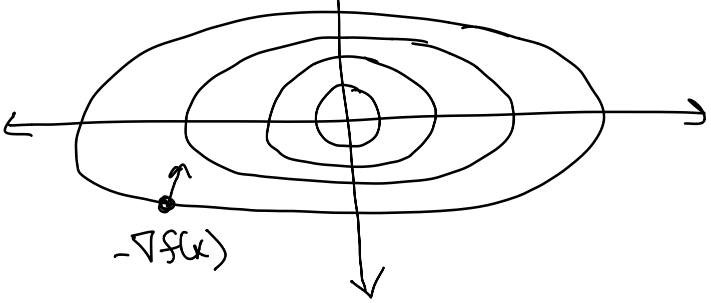
This leads to a simple, natural algorithm for finding a minimum:

Gradient Descent:

```
For t=1 to T
    Set $x_{t+1} = x_t - \eta \nabla f(x_t)$
```

Here $\eta$ is called stepsize

We can see what is happening pictorially



The gradient is $\perp$ to the boundary of the sublevel set, so we are making progress by moving inward

we will merely state some standard guarantees on convergence

def: We say that f is β-smooth if

$$\|\nabla f(x) - \nabla f(y)\| \leq \beta \|x-y\|$$

If f is twice differentiable, equivalently

$$\|\nabla^2 f(x)\| \leq \beta$$

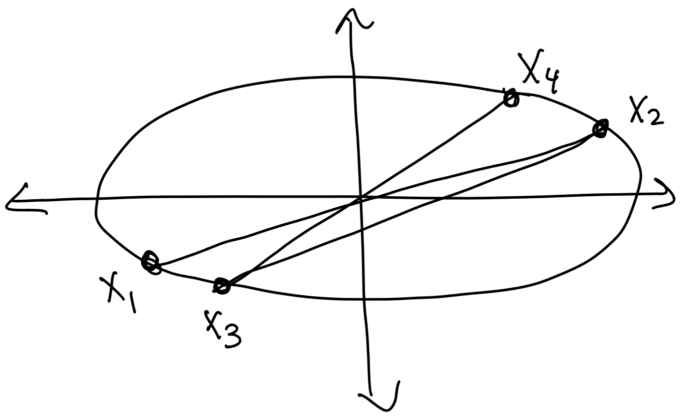
def: We say that f is α-strongly convex if $(y-x)^T \nabla^2 f(x)(y-x) \geq \alpha \|y-x\|^2$

Equivalently

$$f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{\alpha}{2}\|y-x\|^2$$

## Intuition

① we need the gradient to not change too drastically (compared to the stepsize), otherwise



② If we can fit a quadratic under our linear approximation, we know the optimum can't be too far away

Otherwise we would make much slower progress

Main Theorem: If $f$ is $\beta$-smooth and $\alpha$-strongly convex and $\eta \leq \frac{1}{\beta}$ then

$$\|x_{t+1} - x^*\|^2 \leq (1-\eta\alpha)^t \|x_1 - x^*\|^2$$

Note: If $f$ is strongly convex, the minimizer must be unique

Next time: Examples, extensions and nonconvex optimization via neural tangent kernels