Written assignment

1 Lemma 3.1 — Explanation, Context, and Intuition

One-sentence version With a *narrow* one-hidden-layer `tanh` network (width $(s+1)/2$), we can **simultaneously** approximate all odd monomials $x, x^3, \ldots, x^s$ on $[-M, M]$—together with their derivatives up to order k—to any target accuracy $\varepsilon > 0$.

---

1) What does this lemma say?

- **Assumptions**: s is an odd integer ($s \in 2\mathbb{N} - 1$), $k \in \mathbb{N}_0$, and we approximate on the interval $[-M, M]$.
- **Target functions**: all **odd** monomials x^p with $p = 1, 3, 5, \ldots, s$.
- **Accuracy metric**: the $W^{k,\infty}$ norm (i.e., the maximum error over $[-M, M]$ for the function **and** all derivatives up to order k).
- **Guarantee**: For every $\varepsilon > 0$ there exists a **single** shallow `tanh` network of width $(s+1)/2$ such that

$$\max_{\substack{p \le s \\ p \text{ odd}}} \max_{0 \le m \le k} \sup_{x \in [-M,M]} \left| \tfrac{d^m}{dx^m} x^p - \tfrac{d^m}{dx^m} \hat{f}_p(x) \right| \le \varepsilon.$$

  Here, "single network" means **one shared hidden layer** (width $(s+1)/2$) reused for all p. Each $\hat{f}_p$ is obtained by changing only the **final linear head** (output weights). Equivalently, view the construction as **one multi-output network** $\Psi_{s,\varepsilon} : [-M, M] \to \mathbb{R}^{(s+1)/2}$ with

$$\hat{f}_p(x) = e_{(p+1)/2}^{\top} \, \Psi_{s,\varepsilon}(x), \qquad p = 1, 3, \ldots, s.$$

- **Key point**: all odd degrees p **share the same hidden layer**; only the output linear weights change from one p to another.

2) Background / Context

(a) The "cancellation" magic of centered finite differences.

Define the centered p-th finite difference with step h:

$$\delta_h^p[f](x) = \sum_{i=0}^{p} (-1)^i \binom{p}{i} f\Big(x + \big(\tfrac{p}{2} - i\big)h\Big).$$

We sample f at $p+1$ points symmetrically around x and combine them with binomial coefficients. For polynomials, **all terms below degree p cancel out**, leaving only the degree-p contribution.

(b) Taylor expansion.

Let $\sigma = \tanh$. Expanding σ at 0 and inserting into the operator above yields

$$\delta_{hy}^{p} = \sigma^{(p)}(0)\,(hy)^{p} + O\big((hy)^{p+2}\big).$$

Dividing by $\sigma^{(p)}(0)h^p$ we get

$$\widehat{f}_{p,h}(y) = y^p + O\big(h^2|y|^{p+2}\big).$$

(c) Why only $(s+1)/2$ neurons?

Because tanh is **odd** ($\sigma(-t) = -\sigma(t)$), neurons with slopes $\pm a$ can be merged. For a given p we need only $(p+1)/2$ distinct (positive) slopes plus the center. If we prepare the slopes for the **largest** odd degree s, they **cover** all smaller odd p, so the total width is $(s+1)/2$.

3) Construction

For each odd $p \le s$, define

$$\widehat{f}_{p,h}(y) := \frac{1}{\sigma^{(p)}(0)\,h^p} \sum_{i=0}^{p} (-1)^i \binom{p}{i}\, \sigma\Big(\big(\tfrac{p}{2} - i\big)\,h\,y\Big).$$

- **Hidden layer**: neurons $y \mapsto \sigma(a_j y)$ with slopes $a_j \in \{\frac{p}{2}, \frac{p}{2}-1, \ldots, -\frac{p}{2}\} \cdot h$. By oddness, only the positive half plus the middle are needed: $(p+1)/2$ distinct slopes.
- **Output layer**: a fixed linear combination with coefficients $(-1)^i\binom{p}{i}/(\sigma^{(p)}(0)h^p)$.
- **Shared hidden layer**: choose the slope set for $p = s$, namely $\frac{1}{2}h, \frac{3}{2}h, \ldots, \frac{s}{2}h$, as the **common** hidden layer; each smaller p uses a subset of these neurons with its own output weights.

4) Error and derivatives — how do we ensure accuracy up to order k?

We already have $\widehat{f}_{p,h}(y) = y^p + O(h^2|y|^{p+2})$, hence on $|y| \le M$ the function error is $O(h^2)$. Differentiating m times in y (for any $m \le k$) keeps the same structure because differentiation and linear combinations commute, and the same Taylor-plus-cancellation argument applies. Thus

$$\sup_{|y|\le M} \left| \tfrac{d^m}{dy^m} \widehat{f}_{p,h}(y) - \tfrac{d^m}{dy^m} y^p \right| \le C_{k,p,M}\, h^2.$$

Choosing the step: let $B = \max_{p \le s,\ p \text{ odd}} C_{k,p,M}$ and set

$$h = \sqrt{\frac{\varepsilon}{B}} \quad \Rightarrow \quad \text{for all } p \le s,\ 0 \le m \le k,\ \text{the error } \le \varepsilon.$$

5) How large are the weights?

The output coefficients contain $\binom{p}{i}$ and h^{-p}.

- Roughly, $\binom{p}{i} \sim 2^p/\sqrt{p}$ (Stirling's approximation).
- With $h = \Theta(\sqrt{\varepsilon})$, we have $h^{-p} = \varepsilon^{-p/2}$.

So demanding **smaller** error ($\varepsilon \downarrow$) or **higher** degree ($s \uparrow$) inevitably increases weight magnitudes. The paper provides precise upper bounds; the key takeaway is that the growth is controlled and explicit.

6) Two examples (to see the cancellation)

$p = 1$**:**

$$\widehat{f}_{1,h}(y) = \frac{\sigma(\frac{h}{2}y) - \sigma(-\frac{h}{2}y)}{\sigma'(0)\, h} = y + O(h^2y^2).$$

The constant term cancels; the linear term remains.

$p = 3$**:**

$$\widehat{f}_{3,h}(y) = \frac{\sigma(\frac{3h}{2}y) - 3\sigma(\frac{h}{2}y) + 3\sigma(-\frac{h}{2}y) - \sigma(-\frac{3h}{2}y)}{\sigma^{(3)}(0)\, h^3} = y^3 + O(h^2y^5).$$

The first- and second-order contributions cancel; the cubic term remains.

7) Summary

- **Core trick**: centered finite differences + Taylor expansion $\Rightarrow$ lower-order terms cancel, leaving the degree-p term.
- **Error rate**: $O(h^2)$; holds for derivatives up to order k.
- **Step choice**: pick $h \sim \sqrt{\varepsilon}$ to achieve $W^{k,\infty}$ error $\leq \varepsilon$.
- **Network width**: $(s+1)/2$ (odd activation $\Rightarrow$ symmetric slopes can be merged; the slope set for the largest s covers all smaller p).
- **Weight scale**: grows as $\varepsilon \downarrow$ or $s \uparrow$, with explicit bounds in the paper.
- **Bottom line**: a *very narrow* one-hidden-layer `tanh` network can **simultaneously** approximate a whole family of odd monomials (including derivatives).

---

2 Lemma 3.2 — Explanation, Context, and Intuition

One-sentence version On the interval $[-M, M]$, a **single** one-hidden-layer `tanh` network of width $\frac{3(s+1)}{2}$ can **simultaneously** approximate all monomials $y, y^2, \ldots, y^s$ (both odd and even)—together with their derivatives up to order k—to any target accuracy $\varepsilon > 0$. The required weight sizes admit explicit, controlled upper bounds.

1) What does the lemma say?

- **Assumptions**: $s \in 2\mathbb{N} - 1$ (odd), $k \in \mathbb{N}_0$. We approximate on $[-M, M]$.
- **Targets**: all monomials $f_p(y) = y^p$, $p = 1, 2, \ldots, s$.
- **Accuracy metric**: the $W^{k,\infty}$ norm (max error over $[-M, M]$ for the function and all derivatives up to order k).
- **Guarantee**: For any $\varepsilon > 0$ there exists a single shallow `tanh` network $\psi_{s,\varepsilon} : [-M, M] \to \mathbb{R}^s$ of width $\frac{3(s+1)}{2}$ such that

$$\max_{p \le s} \left\| f_p - (\psi_{s,\varepsilon})_p \right\|_{W^{k,\infty}([-M,M])} \le \varepsilon,$$

  where $(\psi_{s,\varepsilon})_p$ denotes the p-th output. Equivalently, it is **one shared hidden layer** with **multiple linear heads** (one per degree p).
- **Key idea**: All **odd** powers are handled by Lemma 3.1. To get **even** powers, use a binomial identity that expresses y^{2n} via *shifted odd powers* $(y \pm \alpha)^{2n+1}$ plus **lower-order even powers**; then define the even approximations **recursively**.

2) Background / intuition: how do we build even powers from odd ones?

The crucial algebraic identity (for any $n \in \mathbb{N}$, $\alpha > 0$) is

$$y^{2n} = \frac{1}{2\alpha(2n+1)}\Big((y+\alpha)^{2n+1} - (y-\alpha)^{2n+1}\Big) - \frac{2}{2\alpha(2n+1)} \sum_{k=0}^{n-1} \binom{2n+1}{2k} \alpha^{2(n-k)+1} y^{2k} \qquad (\star)$$

Reading: an even power $2n$ equals "a difference of two **odd** powers at shifted inputs $y \pm \alpha$" minus "a weighted sum of **lower even** powers".
Why useful? We already know how to approximate **odd** powers (Lemma 3.1), and lower even powers are assumed known by **induction**.

Smallest example ($n = 1$):

$$y^2 = \frac{(y+\alpha)^3 - (y-\alpha)^3}{6\alpha} - \alpha^2.$$

So a quadratic comes from two shifted cubics plus a constant term.

3) Construction — what does the network compute?

- **Odd degrees $p = 1, 3, \ldots, s$:** reuse Lemma 3.1's approximates $\widehat{f}_{p,h}(y)$:

$$(\psi_{s,\varepsilon})_p(y) := \widehat{f}_{p,h}(y) \qquad (p \text{ odd}).$$

- **Even degrees $p = 2n$:** define recursively via ($\star$)

$$(\psi_{s,\varepsilon})_0(y) := 1,$$

$$(\psi_{s,\varepsilon})_{2n}(y) := \frac{\widehat{f}_{2n+1,h}(y+\alpha) - \widehat{f}_{2n+1,h}(y-\alpha)}{2\alpha(2n+1)} - \frac{2}{2\alpha(2n+1)} \sum_{k=0}^{n-1} \binom{2n+1}{2k} \alpha^{2(n-k)+1} (\psi_{s,\varepsilon})_{2k}(y).$$

Why width $\frac{3(s+1)}{2}$?

- For odd powers, Lemma 3.1 needs $(s+1)/2$ distinct slopes after merging $\pm a$ by oddness of `tanh`.
- Even powers require **three input shifts**: $y - \alpha$, y, $y + \alpha$.
- Hence the hidden neurons are of the form

$$\sigma\Big(\big(\tfrac{s}{2} - i\big) h\,(y + \beta)\Big), \quad i = 0, 1, \ldots, \tfrac{s-1}{2},\ \beta \in \{-\alpha, 0, \alpha\},$$

  totaling $3 \times \frac{s+1}{2} = \frac{3(s+1)}{2}$ neurons—exactly the size in the lemma.

4) Error control — define E_p and use induction

Let

$$E_p := \big\|\, f_p - (\psi_{s,\varepsilon})_p \,\big\|_{W^{k,\infty}([-M,M])}.$$

- **Odd**: pick h as in Lemma 3.1, then $E_{2n+1} \le \varepsilon$ for all odd degrees.
- **Even**: plug the *approximate* versions of the right-hand side of ($\star$) and subtract the exact. This yields a **recursive inequality** for E_{2n} in terms of E_{2n+1} and lower E_{2k}. Because $E_{2n+1} \le \varepsilon$ and E_{2k} is nondecreasing in k, induction gives a bound

$$E_{2n} \;\le\; E^*_{2n}(\alpha) \quad \text{(explicit in } \alpha).$$

- **Choose the best** α: one can show that $\alpha = \frac{1}{s}$ minimizes the bound. Substituting it yields $\max_{p \le s} E_p \le \varepsilon$ (after an equivalent rescaling of ε).

Intuition: even powers inherit errors from two shifted odd approximations plus lower even ones; all are controlled, so the total remains $\le \varepsilon$.

5) How large are the weights?

- From Lemma 3.1, odd approximations already entail binomial factors and h^{-p}—weights grow as $\varepsilon \downarrow$ or $s \uparrow$.
- The even construction adds coefficients from ($\star$) and the three shifts. Combining everything gives an **explicit** upper bound (as in the paper); heuristically, think "roughly like $\varepsilon^{-s/2}$ times polynomial/exponential factors in s and M".

6) Example — building y^2 from odd powers

$$y^2 = \frac{(y+\alpha)^3 - (y-\alpha)^3}{6\alpha} - \alpha^2.$$

Steps: (i) obtain $\widehat{f}_{3,h}$ from Lemma 3.1; (ii) evaluate at $y \pm \alpha$; (iii) combine as above and subtract the constant. The total error is a controlled sum of the cubic approximation errors and the constant term.

7) Summary

- **Goal**: a single one-hidden-layer `tanh` network of width $\frac{3(s+1)}{2}$ that approximates $y, \ldots, y^s$ (including derivatives up to k).
- **Technique**: odd via Lemma 3.1; even via identity + recursion.
- **Error**: define E_p; control odd with Lemma 3.1; control even by the recursive bound; choose $\alpha = 1/s$ to get $\max_p E_p \le \varepsilon$.
- **Width**: $(s+1)/2$ shared slopes $\times$ 3 shifts $\Rightarrow \frac{3(s+1)}{2}$.
- **Weights**: explicit growth; increase as $\varepsilon \downarrow$ or $s \uparrow$, but remain controlled by the lemma.

---

3 Unanswered Questions

Context

In binary classification, labels can be wrong, and **where they go wrong** depends on the instance itself: samples that are harder, look more like the other class, or are blurrier are more likely to be mislabeled. As a result, the **observed (noisy) probability** mixes together the **true class probability** and the **mislabeling rate**. Worse, we typically only have very **coarse** noise information (e.g., a **ranking** of which regions are noisier, or **upper/lower bounds** on each sample's noise rate), with no precise noise rates and no clean labels for reference.

Question

If we only know the *ranking* of noise strength (which areas are noisier), can a model—without relying on precise noise rates—still preserve sample *ranking* performance (e.g., AUC) close to what we would obtain with clean labels, or at least preserve ranking consistency up to a monotone transform? If not, does there exist an *unavoidable lower bound* on the resulting ranking bias?