

Gradient Descent from Logarithmic Time to NP-Complete via Data Independence and Sensitive Activations

Jeff Edmonds
York University

October 2026 (corrected version of September 10, 2026)

Abstract

We study the error landscape of neural networks with randomly initialized weights, proving that although suboptimal local minima exist, gradient descent never steps onto the set where the rate matrix $\mathbf{R}$ loses rank (with probability one, for analytic activations); when $MM' \geq D$, every critical point off that set has zero error. Rank alone does not prove convergence. The error does decay to zero when the network is wide ($M \geq D$, f_1 fixed), provided the activation sensitivity α_d stays bounded away from zero. For $M < D \leq MM'$, convergence needs a lower bound on the smallest eigenvalue of $\mathbf{R}$ along training, which we leave open. For a fixed $\mathbf{R}$, the non-ghost error decays at least at rate $\lambda_{\min}^+$, the smallest positive eigenvalue of $\mathbf{R}$. That each data point d decays at its own rate α_d is supported by calculations and simulations, but not proved. We extend the study from wide ($M \geq D$) to overparameterized ($MM' \geq D$) to underparameterized ($MM' < D$) networks. Moreover, widths $M > \text{Rank}(\mathbf{X})$ provide no additional benefit to the training-loss trajectory (we do not study generalization). Here D is the number of data points and MM' is the number of parameters in one fully connected layer of weights $\mathbf{W} \in \mathbb{R}^{M' \times M}$.

At the other extreme, Blum and Rivest [1992]’s seminal work proves that training neural networks is NP-complete. This reduction operates precisely where our “no traps” conditions fail. We prove a $2^{\Omega(\hat{k})}$ lower bound (assuming the Exponential Time Hypothesis) and provide a $D^{O(M'\hat{k}^2)}$ dynamic-programming algorithm for finding $\min_{\mathbf{W}} \max_d \mathcal{E}_d$ when the data representation $\mathbf{X}$ has rank $D - \hat{k}$.

The impressive prior guarantees require either large width (from D^8 neurons in early NTK analyses down to about $D^{3/2}$ parameters for two-layer networks with both layers trained [Song et al., 2021]) or restrict to ReLU (polyhedral methods). We take a third approach. Our model considers any network of the form $f_{\mathbf{W}}(\mathbf{a}) = f_3(\mathbf{W} \cdot f_1(\mathbf{a}))$. Here f_1 and f_3 are arbitrary pre- and post-processing networks that are sufficiently complex (e.g., include at least one fully connected layer of weights wide enough that $\text{Rank}(\mathbf{R})$ can reach $\min(MM', D)$, and nonlinear analytic activation functions). Initially, f_1 and f_3 are either pre-learned or random, but are assumed to have properties that occur with probability one when their parameters are chosen randomly. (This work draws on results from random matrix theory.) Part I proves that the rank of $\mathbf{R}$ is preserved when training all parameters $\theta = \langle f_1, \mathbf{W}, f_3 \rangle$. Part II fixes f_1 and f_3 and studies per-point rates when training on only one layer $\mathbf{W}$.

In conclusion, the complexity and randomness of f_1 and f_3 , the sensitivity α_d , the rank deficit k of the rate matrix $\mathbf{R}$, and the rank deficit $\hat{k}$ of the data representation $\mathbf{X}$ act as faucets that smoothly interpolate between tractable and intractable regimes.

1 Introduction

A neural network with a trillion parameters cannot afford more than a logarithmic number of iterations per training example. Empirically, gradient descent on highly non-convex error landscapes frequently reaches near-zero training error—and sometimes it does not. We are pleased to add to the extensive theoretical work rapidly advancing this understanding.

Previous Methods: Methods for proving that gradient descent decays the error geometrically fall into two families:

- To bypass non-convexity, the very influential work of Jacot et al. [2018] introduced the Neural Tangent Kernel (NTK), which linearizes the training dynamics around initialization. A large body of follow-up work has refined and extended this approach.
- In a breakthrough result, Nguyen [2021] uses polyhedral algebraic methods to exploit the piecewise-linear structure of ReLU.

Our new method includes:

- **No Kernel:** While the NTK ignores the data itself except via a kernel $\mathbf{K} \in \mathbb{R}^{D \times D}$ with entries $K_{dd'} = \phi(\mathbf{x}_d)^T \phi(\mathbf{x}_{d'})$, our results work directly with the rate matrix $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$ and the Gram matrix $\mathbf{X}^T \mathbf{X}$ of the actual data representations.
- **Multiple Levels:** Our model considers any network $f_{\mathbf{W}}(\mathbf{a}) = f_3(\mathbf{W} \cdot f_1(\mathbf{a}))$, where $\mathbf{W} \in \mathbb{R}^{M' \times M}$ is a fully connected layer and f_1 and f_3 are arbitrary pre- and post-processing networks that are sufficiently complex (e.g., include at least one fully connected layer of weights and nonlinear piecewise-analytic activation functions).
- **Any Activations:** While most restrict to ReLU or sigmoids, we consider any non-linear piecewise-analytic activation function—covering ReLU, sigmoid, tanh, swish, and softplus. The rank-preservation result of Part I, however, needs analytic activations, so it does not cover ReLU.
- **Local Sensitivity:** Gradient descent can stall if it moves into territory where the activation function’s (e.g. sigmoid or ReLU) slope is near zero. Define $\alpha_d = \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2$ to be the point-wise sensitivity of the network with respect to the current pre-activation values $\mathbf{z}_d$ for data point d . Our convergence results depend on these being large and our NP-completeness results depend on these being small.
- **Error Landscape and Traps (Part I):** Though Ding et al. [2022] proved suboptimal local minima exist, we prove that gradient descent never steps onto these traps: it never enters the set where the rate matrix $\mathbf{R}$ loses rank, and when $MM' \geq D$ every critical point off that set has zero error (under mild conditions, with probability one over random initialization). This alone does not prove convergence. In a small example, gradient descent keeps $\mathbf{R}_t$ at full rank at every iteration yet converges to a suboptimal local minimum, because the smallest eigenvalue of $\mathbf{R}_t$ tends to zero (Remark 2). In the wide case ($M \geq D$, f_1 fixed), that eigenvalue stays at least $\lambda_{\min}(\mathbf{X}^T \mathbf{X}) \alpha_{\min}$, so there the error does decay to zero (Remark 1). For $M < D \leq MM'$, a lower bound on it along training is our central open problem. When underparameterized $MM' < D$ and $\mathbf{R}$ is held fixed, the non-ghost error decays to zero; however, $k = D - MM'$ dimensions of ghost error do not decay. Given the rank constraint, this could be called optimal. We also give an engineering/quantified version of the result.
- **Tracking Point-wise Decay (Part II):** While most previous results track the total error $\mathcal{E}$, we track the point-wise error $\mathcal{E}_d = (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2$ which is ultimately what we

care about.

- **Parametrization:** We extend the study from wide ($M \geq D$) to overparameterized ($MM' \geq D$) to underparameterized ($MM' < D$) networks. Here D is the number of data points and MM' is the number of parameters in one fully connected layer of weights $\mathbf{W} \in \mathbb{R}^{M' \times M}$.
- **Removing Redundant Rows:** A common intuition suggests that overparameterization improves optimization by expanding the basins of attraction and creating additional paths to global minima. We prove that this is not the case: any redundant ($M > \hat{D}$) network is a “shadow” of the non-redundant ($M = \hat{D}$) case—same critical points, same basins, same convergence rates. This concerns the training-loss trajectory only; we do not study generalization.
- **Hardness of Training (Part III):** Blum and Rivest [1992]’s seminal work proves that training neural networks is NP-complete. We tighten this by analyzing small rank deficits $\hat{k} = D - \text{Rank}(\mathbf{X})$. We prove a $2^{\Omega(\hat{k})}$ lower bound¹ and provide a $D^{O(M'\hat{k}^2)}$ dynamic-programming algorithm for finding $\min_{\mathbf{W}} \max_d \mathcal{E}_d$, showing the problem is polynomial for each fixed $\hat{k}$ and M' (the class XP). We do not know whether it is fixed-parameter tractable. The complexity of finding $\min_{\mathbf{W}} \sum_d \mathcal{E}_d$ is open. The NP reduction operates precisely where our “no traps” conditions fail—non-random f_3 with $\alpha_d = 0$, narrow width ($MM' < D$), and rank-deficient $\mathbf{X}$ —confirming that the continuous boundary between tractable and intractable aligns across both convergence and complexity.

2 Paper Organization

- **Introduction:** Notation; Table of Results; Architecture; Related Work
- **Part I (Suboptimal Local Minima Exist but Are Avoided):**
Generalizes Nguyen and Hein [2017]’s landscape result, showing that although suboptimal local minima exist, gradient descent never steps onto them (rank preservation). This gives convergence in the wide case; for $M < D \leq MM'$ convergence is open.
 - **Wide Networks ($M \geq D$):** Repeats proof for NH’s trivial case. (S7)
 - **Rank Preservation (any MM'):** Proves the result. (S8)
 - **Engineering/Quantified Bounds (Robustness)** (S9)
- **Part II (Per-Point Convergence):**
This studies per-point $e_d = f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*$ convergence rates for gradient descent: some are proved (mostly for a fixed rate matrix $\mathbf{R}$), others are conjectured and tested by simulation. These results are classified into:
 - **Gradient Descent:** This is a gentle explanation of the matrix magic. (S10)
 - **Spectral Results:** Eigendecomposition of the rate matrix $\mathbf{R}$ (S11)
 - **Simulations:** To understand $\mathbf{R}$ better (S12)
 - **Signal vs Noise Results:** Bounds the effect of noise (off-diagonals of $\mathbf{R}$). (S13&14)
 - **Random Walks:** A study of what goes wrong (S15)
 - **Random Network Structures:** Properties of random f_1 and f_3 (S16)
 - **Extensions and Generalizations:** Between the Lazy (NTK) and Feature-Learning (S17.1); Equal $\hat{\lambda}_i$; Ignoring Stalled Points; Worst case Weights; Second-Order Ef-

¹ Assuming the Exponential Time Hypothesis (ETH)

- fects; $\langle \alpha, \beta \rangle$ -PL Sensitivity; Removing Redundant Rows; (S17)
- **Part III (NP-Completeness & DP Algorithm):**
 - **NP-Completeness Reduction** (S18)
 - **Increasing the Arity M' of the AND** (S19)
 - **Algorithm for the $\hat{\mathbf{k}} = \mathbf{D} - \hat{\mathbf{D}}$ Rank Deficit Problem** (S20)

Each part opens with a detailed roadmap of its sections.

3 Notation

Entity	Individual	Collection	Dims $(2^{nd} \text{ col})^{3^{rd}}$
Model of Network:	$f_{\mathbf{W}}(\mathbf{a}) := f_3(\mathbf{W} \cdot f_1(\mathbf{a}))$ • General, except that it must include at least one fully connected layer of trainable weights $\mathbf{W}$. • There may be other structures of any kind, but our results do not consider training these weights. • any activation function that is differentiable except at finitely many points (ReLU, sigmoid, ...) • D data points & M width		
Raw Data Input	$\mathbf{a}_d \quad d \in [\text{Data Point}] = [D]$		$(\mathbb{R}^N)^D$
Early Layers of NN	$f_1: \mathbb{R}^N \rightarrow \mathbb{R}^M$ (fixed/arbitrary)		
Post- f_1 Representation	$\mathbf{x}_d := f_1(\mathbf{a}_d)$ with features m	$\mathbf{X}$	$(\mathbb{R}^M)^D$
Weights	$w_{MM'}$	$\mathbf{W}$	$(\mathbb{R})^{M' \times M}$
Pre- f_3 Representation	$\mathbf{z}_d := \mathbf{W} \mathbf{x}_d$ with features m'	$\mathbf{Z}$	$(\mathbb{R}^{M'})^D$
Later Layers of NN	$f_3: \mathbb{R}^{M'} \rightarrow \mathbb{R}$ (fixed/arbitrary)		
Network Output	$f_{\mathbf{W}}(\mathbf{a}_d) := f_3(\mathbf{z}_d)$		$(\mathbb{R})$
Target	y_d^*		$(\mathbb{R})$
Point-wise error:			
Pre-squared	$e_d := f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*$	$\mathbf{e}$	$(\mathbb{R})^D$
Squared	$\mathcal{E}_d := (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2$	$\mathcal{E} := \sum_d \mathcal{E}_d$	$(\mathbb{R})$
Non-Ghost	$\hat{\mathcal{E}}_d$ of fraction P_{dd}		
Rate	$r_d \in \{\lambda_{\min}^+, \alpha_{\text{coll}}, \alpha_d\}$ such that $\mathbb{E}[\mathcal{E}_{d(t+1)}] = (1 - 2\eta r_d)^2 \mathbb{E}[\mathcal{E}_{dt}]$		
Target error size	ϵ		
Iteration	Added index $t \in [T]$ Initial values indexed with 0.		
About f_3			
Gradients	$\mathbf{g}_d := \nabla_{\mathbf{z}} f_3(\mathbf{z}_d) := \nabla_{\mathbf{z}} e_d$	$\mathbf{G}$	$(\mathbb{R}^{M'})^D$
Sensitivity/Slope	$\alpha_d := \ \mathbf{g}_d\ ^2 \quad \bar{\alpha} := \text{Avg } \alpha_d$		$(\mathbb{R})$

Collective	$\alpha_{\text{coll}} := \frac{\sum_d \alpha_d^2 \mathcal{E}_d}{\sum_d \alpha_d \mathcal{E}_d} \geq \alpha_{\min}^2$		
Rate Matrix	$r_{dd'} := (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'})$	$\mathbf{R}$	$\mathbb{R}^{D \times D}$
$\mathbf{e}_t = \prod_t (\mathbf{I} - 2\eta \mathbf{R}_t) \mathbf{e}_0 \quad e_{dt} = \sum_i (1 - 2\eta \lambda_i)^t v_{di} e_{\parallel i0} \quad T \leq O(\lambda_{\min}^{+1} \log(\mathcal{E}_0/\epsilon))$			
Channel i	$i \in [D]$		$[D]$
Eigenvectors	v_{di}	$\mathbf{V}$	$(\mathbb{R})^{D \times D}$
Eigenvalues	$\lambda_i \quad \lambda_{\min}^+ := \text{smallest non-zero}$	Λ	$(\mathbb{R})^D$
Ghost channel i'	$\lambda_{i'} = 0 \quad i' \in [D' + 1, \dots, D]$		
Spectral Error	$e_{\parallel i} := \mathbf{v}_i^T \mathbf{e}$ the component of the error in the $\mathbf{v}_i$ direction.		$(\mathbb{R})$
Squared	$\mathcal{E}_{\parallel i} := e_{\parallel i}^2$	$\mathcal{E} = \sum_i \mathcal{E}_{\parallel i}$	$(\mathbb{R})$
Channel Independence	$\mathbb{E}[e_{\parallel i}^2] = \sigma^2$ & $\mathbb{E}[e_{\parallel i} e_{\parallel j}] = 0$ for $i \neq j$		
Non-ghost error	$\hat{\mathcal{E}}_d := \left(\sum_{i=1}^{D'} e_{\parallel i} v_{di} \right)^2$ (point-wise)		
Fraction non-ghost	$P_{dd} = \sum_{i=1}^{D'} v_{di}^2$		
Signal vs Noise	$\alpha_d e_d$ vs $\xi_d := \sum_{d' \neq d} r_{dd'} e_{d'}$		
$\mathbf{R}$ Rank	D'		
Rank Deficit	$k := D - D'$		
GD step size	$\eta > 0$ (small)		
Bad set	$\mathcal{B} := \{\text{Rank}(\mathbf{R}) < D'\}$		
About $\mathbf{X}$	$x_{md} \quad m \in [\text{feature}] = [M]$	$\mathbf{X}$	$(\mathbb{R})^{M \times D}$
Rank	$\hat{D} := \text{Rank}(\mathbf{X})$		$(\mathbb{R})$
Rank Deficit	$\hat{k} := D - \hat{D}$		$(\mathbb{R})$
Eigenvector of $\mathbf{X}^T \mathbf{X}$	$\hat{\mathbf{v}}_i$	$\hat{\mathbf{V}}$	$(\mathbb{R}^D)^D$
$\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}} \mathbf{H}$	Randomly rotated eigenvectors $\hat{v}_{di} \approx \pm \frac{1}{\sqrt{D}}$		
Eigenvalue	$\hat{\lambda}_i$		$(\mathbb{R})^D$
Threshold Constant	$\rho := \frac{D}{MM'}$		
Kronecker Delta	$\delta_{ij} := \begin{cases} 1 & \text{if } i=j \\ 0 & \text{else} \end{cases}$ $\delta_{ijk} := \begin{cases} 1 & \text{if } i=j=k \\ 0 & \text{else} \end{cases}$		$(\mathbb{R})$

4 Our Table of Results

	Sensitive Activations (Real)	Insensitive (Boolean)
Activation Functions: Sigmoid, ReLU, Tanh, Swish, Softplus	Wide: The activations are wide enough that practitioners keep each activation value z within its middle range.	Narrow: The network has many sigmoids that are so narrow that they are effectively Boolean thresholds.

² $\alpha_{\min} := \min_d \alpha_d$, $e_{\max} := \max_d |e_d|$, and so on.

Slope/Sensitivity:	Steep: • This assures us that the readout f_3 responds strongly to changes in its input, measured by $\alpha_d = \ \nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\ ^2$ being bounded away from zero.	Flat: • Output changes abruptly with flat regions ($\alpha_d \approx 0$) in between.
Computational Complexity:	Linear Programming: • Linear relaxation allowing real-valued solutions which is typically polynomial-time.	Integer Programming: • Discrete/Boolean problems may admit dynamic programming solutions but are generally NP-complete.
Landscape: • Partition weight space $\mathcal{W}_{\geq \alpha} = \{\mathbf{W} \mid \forall d, \alpha_d \geq \alpha\}$	Flow to the Bottom: • Although suboptimal local minima exist, gradient descent never steps onto them ($\mathbf{R}_t$ keeps full rank). The error decays to zero when $M \geq D$; for $M < D \leq MM'$ this needs a floor on $\lambda_{\min}(\mathbf{R}_t)$, which is open (Theorem 1)	Local Minimum: • $\mathcal{W}_{< \alpha}$ can harbor exponentially many local minima that make optimization hard. (Theorem 4.2)
Spectral Results: • Errors decay with $\mathbf{e}_t = \prod_t (\mathbf{I} - 2\eta \mathbf{R}_t) \mathbf{e}_0$ $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$ into channels with rates λ_i • Channels with $\lambda_i = 0$ are called <i>ghost</i> and do not decay under the current $\mathbf{R}$.	Point-wise Error: • At iteration t, the non-squared error $e_d = f\mathbf{w}(\mathbf{a}_d) - y_d^*$ has decayed to $e_{dt} = \sum_i (1 - 2\eta \lambda_i)^t v_{di} e_{i0}$. • Rate $r_d \in [\lambda_{\min}^+, \alpha_d]$ (conjectured). • $O(\lambda_{\min}^{+ -1} \log(\mathcal{E}_0/\epsilon))$ iterations (fixed $\mathbf{R}$). • Confirmed with simulations. (Section 11 & 12)	GD Stalls: • When $\alpha_d \approx 0$, the per-point rate vanishes and descent stalls (Section 17.3).
Random: Results require randomness in the input to gradient descent.	Random f_1 & $\mathbf{x}_d$ Random f_3 & $\mathbf{g}_d$ • Hence, random rate matrix $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$	
Random Walks: • Signal $\alpha_d e_d$ steady • Noise ξ_d has random sign; should grow with $\sqrt{T}$. • t dependence handled.	Fails: • Ghost gives $\text{sign}(\xi_d) \approx \text{sign}(e_d)$. (Section 15)	

A Mix:		Partition Data Points: • Active: Reasonable sensitivity $\alpha_d \geq \alpha_A$ • Frozen: Flat sensitivity $\alpha_d \leq \alpha_F$ • Don't interact if $\alpha_F \ll \alpha_A$ • Theorem 2.7.2	
Results sorted by: • $M \& M'$ are network widths • MM' is the number of trainable parameters in the weights matrix $\mathbf{W} \in \mathbb{R}^{M' \times M}$. • D is the number of data points.			
Prior: (Du et al. & Allen-Zhu et al.) • $M \geq D^6$	Convergence rates similar to ours		
Coherent Regime: • $M \geq \Omega(D^2)$	Random f_1 & Arbitrary f_3: • Negligible Noise: $r_d = \alpha_d$ • Only while the errors are comparable ($e_{\max}/e_{\min} = O(1)$).		
Wide Network: • $M \geq \Omega(D)$ • The network's width exceeds the number of data points.	Random f_1 & Random f_3: • Negligible Noise: $r_d = \alpha_d$ • Only while $e_{\max}/e_{\min} = O(1)$ and the $\mathbf{g}_d$ stay nearly orthogonal (known at initialization). Random f_1 & Arbitrary f_3: • Managed: assuming a coupling hypothesis on the noise (not yet derived from the random model), $\overline{\alpha\mathcal{E}}$ decays at rate $\geq \frac{1}{2}\alpha_{\text{coll}} \geq \frac{1}{2}\alpha_{\min}$. Theorem 2.3 & 2.4	Independent Control: • Linear independence gives the data points independent control so that each can reach their target. • The error landscape does not contain nonzero local minima. • Solvable via linear algebraic systems. • Theorem 4.1	
Small Rank Deficit • $M = D - \hat{k}$	Random f_1 & Linear f_3: • $\hat{k}$ ghost spectral errors $\lambda_i = 0$ that never decay.	Algorithmic Bridge (XP): • $2^{\Omega(\hat{k})}$ vs $D^{O(M'\hat{k}^2)}$ time for finding $\min_{\mathbf{W}} \max_d \mathcal{E}_d$ • Complexity $\min_{\mathbf{W}} \sum_d \mathcal{E}_d$ open. • Theorem 4.3 & 4.4	

Overparameterized: • $MM' \geq \Omega(D)$ • The total number of trainable parameters exceeds the number of data points.	Random f_1 & Random f_3: • Managed: conditional, as above • No ghost errors • $\mathbf{R}_t$ keeps full rank (Part I); convergence is open	Conflict: • Linear dependence between the data points causes conflict and nonzero local minima. NP-Complete: • We adjust Blum and Rivest’s 3-SAT NP-Complete reduction to fit into our framework. • Theorem 4.2
Underparameterized • $\text{Rank}(\mathbf{R}) = MM' = D - k$	Random f_1 & Random f_3: • k ghost spectral errors $\lambda_i = 0$ that never decay.	
Redundant Width: • $M > \hat{D}$	No Help: A common intuition suggests that widths $M > D$ give “Basin Expansion,” increasing the “number of paths” to the global minimum. We show that the additional weights $\mathbf{W}_\perp$ are frozen: no functional gain, no additional optimization paths, no basin expansion. The effective search space is identical to the $M = \hat{D}$ case (training loss only; generalization is not studied). • Section 17.7	More Likely Data Independence: Extra width increases the likelihood that the data representations are linearly independent. Once this is achieved, extra width is not helpful for training.
Exponential Density • $D = 2^{o(M)}$	Near-Orthogonal Stability: • Within low M dimensional space, the network can have an exponential number of learned/random near-orthogonal feature vectors $\phi_j \in \mathbb{R}^M$, as long as each data point representation $\mathbf{x}_d \in \mathbb{R}^M$ is a linear combination of only a few of them.	

5 The Model/Architecture of Our Network

Finding new neural network architectures has been hugely fruitful. Our model considers any network that includes at least one fully connected layer of trainable weights $\mathbf{W}$. See Figure 1.

General Model: We include the classic model consisting of multiple layers of alternations between fully connected weights and an activation function. We consider any piecewise-analytic activation function (ReLU, sigmoid, tanh, swish, softplus). We also include much

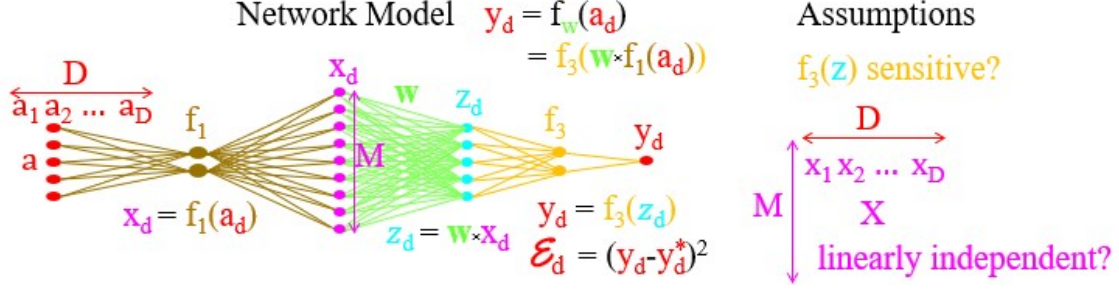


Figure 1: Architecture $f_W(a) = f_3(W \cdot f_1(a))$

fancier structures like transformers. The only restriction is that our network needs to contain at least one fully connected layer $W \in \mathbb{R}^{M' \times M}$.

Three Layer Decomposition: Our network architecture is of the form $f_W(a_d) = f_3(W f_1(a_d))$. Here f_1 denotes all layers before the trained weights W and f_3 all layers after. Note that when modeling the standard alternating-layer network, this assumes f_1 ends in a non-linear activation function and f_3 begins with one.

More detail: A fixed *feature map* f_1 produces data representations $x_d = f_1(a_d) \in \mathbb{R}^M$, a weight matrix $W \in \mathbb{R}^{M' \times M}$ maps these to pre-activations $z_d = W x_d$, and a differentiable fixed readout f_3 produces the network output. The total squared error is $\mathcal{E} = \sum_{d=1}^D (f_3(z_d) - y_d^*)^2$.

A Limit on Learning: Part I, which proves that gradient descent decays the non-ghost error to zero, trains on all of the network. In contrast, Part II, which bounds decay rates, trains on only one layer W at a time so that the rate matrix R is easier to define. Training other structures (eg transforms) is beyond the scope of the paper. See Section 17.1 for where this regime sits between lazy training and feature learning.

One Layer of Weights at a Time: Our per-point convergence analysis (Part II) considers one fully connected layer W_ℓ at a time. This, however, is not a limitation because the effect of updating all the layers simultaneously by linearity of the gradient is $\frac{d\mathcal{E}}{dt} = \sum_\ell \frac{\partial \mathcal{E}}{\partial W_\ell} \frac{dW_\ell}{dt}$, where each summand is our single-layer result applied to W_ℓ .³

Pre-Processing: The purpose of the $x_d = f_1(a_d) \in \mathbb{R}^M$ stage is to extract M interesting linearly independent features from the input. This stage could represent a pre-trained, frozen backbone or transfer-learning pipeline. Our analysis begins at x_d .

Requirements:

- **Convergence Results:**

- **Sufficiently Complex:** For our positive results, we require f_1 and f_3 to contain at least one fully connected layer and non-linear piecewise-analytic activation functions. For example, f_1 could map an input consisting of a single real number a_d to one layer of activations followed by a fully connected layer of weights (e.g., $W' \in \mathbb{R}^{M \times M}$).
- **Random and Fixed:** We fix f_1 and f_3 and assume they have properties that occur with probability one when chosen randomly.

³Over multiple iterations these changes in the layers interact. In fact, Section 12 considers decomposing the ghost error by changing R by changing f_1 and f_3 .

- **f_3 Sufficiently Sensitive:** We need that it responds sensitively to changes in its input, measured by $\alpha_d = \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2$. We do not perform this analysis ourselves because the sensitivity of f_3 is already well-studied.
- **Parameterization Regimes:** The trainable layer $\mathbf{W} \in \mathbb{R}^{M' \times M}$ has MM' parameters and the number of data points is D .
 - * **Wide Network:** $M \geq D$
Needed for making the data representation $\mathbf{x}_d$ linearly independent.
 - * **Overparameterized Network:** $MM' \geq D$
Needed for the rate matrix $\mathbf{R} \in \mathbb{R}^{D \times D}$ to have rank D .
 - * **Underparameterized Network:** $MM' < D$.
This implies $k = D - \text{Rank}(\mathbf{R}) = D - MM'$ ghost errors.
 - * (Theorem 2.6)
- **NP Complete Results:**
 - **f_3 is Neither Random nor Complex:** f_1 is the identity mapping and f_3 is the threshold of a big AND.
 - **f_3 Flat:** The gradient of f_3 is effectively $\alpha_d = 0$ because the activation functions are so narrow that they look like Boolean thresholds.
 - **Not Wide:** $M = \frac{1}{3}D$. Rank deficits $\hat{k} = D - \text{Rank}(\mathbf{X}) = D - \frac{1}{3}D$.

6 Related Work

The question of *when* and *why* gradient descent can train neural networks has a long history, and our work builds on ideas from several communities. We organize the literature into various threads and explain how our results relate to each.

Linear Networks: It is true that the function computed by a deep completely linear network $\mathbf{W}_{\text{eff}} = \mathbf{W}_L \cdots \mathbf{W}_1 \mathbf{x}$ is just one big linear map. However, if you optimize over each $\mathbf{W}_\ell$ separately, the loss as a function is a polynomial of degree $2L$ — degree L from the product, squared by the loss. This is not convex and has saddle points. Kawaguchi [2016] proves that these networks have no suboptimal local minima. This is a clean and influential result relying on the algebraic structure of linear maps.

Error Landscape and Local Minima: A foundational line of work, which culminates with Nguyen and Hein [2017], proves that with wide networks ($M \geq D$), with analytic activations, almost all critical points ($\nabla_{\theta} \mathcal{E}(\theta_*) = \mathbf{0}$) are global minima ($\mathcal{E}(\theta_*) = 0$).^{4 5} Ding et al. [2022] proved that this “almost all” is tight because for deep networks (two or more hidden layers) with smooth activations, such suboptimal local minima exist regardless of width.

Our Contribution: We prove that gradient descent never steps onto such traps ($\mathbf{R}_t$ keeps full rank, with probability one). In the wide case (f_1 fixed) the error then decays to zero. We relax the wide ($M \geq D$) requirement to overparameterized ($MM' \geq D$) for rank preservation; whether the error then decays to zero is open. When underparameterized, the non-ghost error decays to zero under a fixed $\mathbf{R}$.

⁴Requires either the current data representation $\mathbf{X}_* \in \mathbb{R}^{M \times D}$ or the Hessian $\mathbf{H}(\theta) := \nabla_{\theta}^2 \mathcal{E}(\theta) \in \mathbb{R}^{|\theta| \times |\theta|}$ to be nondegenerate.

⁵Nguyen and Hein [2018] extended this to convolutional architectures. Jiang [2019] proposed a framework mapping neural networks to a “Canonical Space” where optimization becomes convex. However, the analysis relies on a Fourier ϵ -approximation that can require width exponential in the input dimension.

Six Papers: Here we are going to compare six of the many papers in this area: Jacot et al. [2018] (Chizat and Bach [2018]), Du et al. [2019], Allen-Zhu et al. [2019], Arora et al. [2019], Nguyen [2021], and our work. These results fall into three methodological families: (1) NTK perturbation analysis (Jacot⁶, Du, Arora, Allen-Zhu) which linearizes around initialization; (2) polyhedral algebraic methods (Nguyen), which exploit the piecewise-linear structure of ReLU; and (3) direct spectral analysis (this work), which tracks per-point sensitivity through smooth activations.

Architecture of Network: All papers consider multiple layers of alternations of fully connected weights and an activation function. Jacot and ourselves consider any activation function that is differentiable except at finitely many points, while Du, Arora, Allen-Zhu, and Nguyen restrict their attention to ReLU so that the entire network is piecewise linear. Du and Arora only handle two levels ($\mathbf{W}_1$ -ReLU- $\mathbf{W}_2$ -Sum) while Jacot, Allen-Zhu, Nguyen, and ourselves consider arbitrary depth L .

Our Contribution: Our model $f_{\mathbf{W}}(\mathbf{a}_d) = f_3(\mathbf{W}f_1(\mathbf{a}_d))$, in addition to including theirs, includes much fancier structures like transformers. Our per-point analysis trains one layer $\mathbf{W}$ at a time; the landscape results extend to all-layer training. Unlike Nguyen, we are not restricted to ReLU activations.

Readout Sensitivity: Gradient descent can stall if it moves into territory where the activation function’s (e.g. sigmoid or ReLU) slope is near zero. All papers must assume that the activation sensitivities α_d remain bounded away from zero. Nguyen’s result depends on $\sigma_{\min}(F_{L-1})$, the smallest singular value of the feature matrix F_{L-1} , whose rows are the data representations at the penultimate hidden layer.

Error Decay: With such big networks, the number of gradient descent iterations⁷ really must be logarithmic. The NTK analyses above all bound total loss decay at rate $r = \lambda_{\min}(\mathbf{K})$, giving $\mathcal{E}(t) \leq (1 - \eta\lambda_{\min})^t \mathcal{E}_0$ and hence $O(\lambda_{\min}^{-1} \log(\mathcal{E}_0/\epsilon))$ iterations.

Our Contribution: Our results separate their one global sensitivity $\sigma_{\min}(F_{L-1})$ and rate $\lambda_{\min}(\mathbf{K})$ into separate per-point α_d values. Here $\alpha_d := \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2$ is the sensitivity of the final output of the neural network with respect to the current activation values $\mathbf{z}_d$ for data point d . Decay of error $\mathcal{E}_d := (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2$ requires $\Theta(\alpha_d^{-1} \log(\mathcal{E}_0/\epsilon))$ iterations (conjectured, supported by simulations). Some of our results, however, resort back to single numbers like $\bar{\alpha}$, $\alpha_{\min}$, or $\lambda_{\min}^+$.

Neural Tangent Kernel (NTK): To bypass non-convexity, the very influential work of Jacot et al. [2018] introduced the concept of the NTK, and (Jacot, Du, Arora, Allen-Zhu) use it. For each data point, let $\mathbf{x}_d$ be either the data vector itself or some hidden representation deeper in the network. Let $\phi(\mathbf{x}_d) \in \mathbb{R}^M$ be some fixed mapping of it to M features. Define the kernel $\mathbf{K} \in \mathbb{R}^{D \times D}$ with entries $K_{dd'} = \phi(\mathbf{x}_d)^T \phi(\mathbf{x}_{d'})$ to be the dot product between any pair of such data points. Beyond this, the data itself is ignored. The wider M , the more the network under this fixed kernel evolves like a linear model.

Our Contribution: Instead of considering the kernel $\mathbf{K}$ of a higher dimensional space, we work directly with rate matrix $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X}) \in \mathbb{R}^{D \times D}$ for our data.

⁶Chizat and Bach [2018] analyzed the same infinite-width regime as Jacot et al. [2018] from an (optimal transport) perspective, proving global convergence.

⁷Here an “iteration” is assumed to use all of the data once.

Spectral Decomposition: They decompose their kernel into $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ with eigenvectors $\mathbf{v}_i$ and eigenvalues $\lambda_i \geq 0$. (These are different from ours.) A follow-up by Jacot et al. [2019] showed that in the infinite-width NTK regime the eigenvalues λ_i determine the eigenvalues of the Hessian matrix of the loss during training, i.e. the second derivatives with respect to weights, telling you which directions of the error landscape are steep vs. flat.

Spectral Decomposition of Error: This is classical for linear systems and it is done in (Jacot, Du, Arora, Allen-Zhu, ourselves). The error of gradient descent can then be decomposed $\mathcal{E} = \sum_i \mathcal{E}_{\parallel i}$ into spectral components $\mathcal{E}_{\parallel i} = (\mathbf{v}_i^T \mathbf{e})^2$ where $\mathbf{e} \in \mathbb{R}^D$ is the error vector with $e_d = (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)$ for data point $\langle \mathbf{a}_d, y_d^* \rangle$ and weights $\mathbf{W}$.

Tracking Separate Decay: Most of the results track the number of iterations for the total error $\mathcal{E}$ to decay, with time proportional to $\frac{1}{\lambda_{\min}}$ with respect to the smallest eigenvalue. Instead, (classical linear systems, Jacot, Arora, ourselves) track each spectral error component $\mathcal{E}_{\parallel i}$ separately. It is common to say that gradient descent has a separate “channel” with rate λ_i along which this error $\mathcal{E}_{\parallel i}$ should decay with required time proportional to $\frac{1}{\lambda_i}$. Each channel has its own faucet, and the faucet closes smoothly as λ_i shrinks.

Avoiding Eigendecompositions: In contrast, Nguyen [2021] avoids eigendecomposing the NTK kernel: he lower-bounds $\mu_{\min}^+(\mathbf{K})$ via $\sigma_{\min}(F_{L-1})$ and uses Lipschitz bounds on ReLU to show this doesn’t drift far from initialization, obtaining total-loss convergence without eigendecomposition.

Our Contribution: Like the previous, we have one section considering the eigendecomposition of our rate matrix $\mathbf{R}$ into separate rate λ_i channels. However, because of the lack of results about this decomposition, Nguyen and our main results avoid this. Instead, we bound the effect of the noise (off-diagonals of $\mathbf{R}$) on the signal α_d (diagonal entries). When $M \geq \Omega(D)$ and the errors are comparable in size, the noise is negligible and hence with high probability the spectral error $\mathcal{E}_{\parallel i}$ aligns with the data point error $\mathcal{E}_d = (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2$ which is ultimately what we care about. This leaves only the point-wise decay rate $r_d = \alpha_d$. When $MM' \geq \Omega(D)$, under a coupling hypothesis that we have not yet derived, the noise is manageable and we are able to track the collective decay.

Ghost Errors: A spectral error component is said to be *ghost* if $\lambda_i = 0$, because its decay time $\frac{1}{\lambda_i}$ is infinite. All prior work assumes $\forall i \lambda_i > 0$.⁸

Our Contribution: It is easy to assume that ghost errors appear when the width $M < D$ is less than the number of data points. The intuition being that here the data representation $\mathbf{X} \in \mathbb{R}^{M \times D}$ is not full rank and hence the Gram matrix $\mathbf{X}^T \mathbf{X}$ has zero eigenvalues $\hat{\lambda}_i$. We, however, show that with random f_1 and f_3 , they do not appear until the number of trainable parameters $MM' < D$ is less than the number of data points. This is because, surprisingly, the rate matrix $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$ whp has rank $D' = \min(D, MM')$ (Theorem 2.6.3).

Underparameterized: When $MM' < D$, the rate matrix $\mathbf{R}$ has rank at most MM' , leaving $k = D - \text{Rank}(\mathbf{R})$ ghost channels with $\lambda_i = 0$ whose errors do not decay under fixed $\mathbf{R}$. The remaining non-ghost errors decay rapidly — faster than in the overparameterized case, since fewer channels share the total rate budget $\sum_d \alpha_d$. Our simulations

⁸For example, Arora et al. require $M \propto \lambda_{\min}^{-4}$.

show that when $\mathbf{R}_t$ varies over iterations, the rotating null space can gradually absorb ghost errors as well.

Table of Results: The following lists the convergence guarantees for gradient descent on neural networks. Unless otherwise stated, all achieve geometric convergence: $O(\log(1/\epsilon))$ iterations. D = number of training points (called n in most papers).

	L	M	Act.	Res. ⁹	Comment
Jacot et al. [2018]	L	$\rightarrow \infty$	any	$\mathcal{E}_{\parallel i}$	Foundational. The NTK insight created an entire sub-field.
Du et al. [2019]	2	$\frac{D^6}{\lambda_{\min}^4}$	ReLU	$\mathcal{E}$	First to make NTK work at finite width + ReLU. Shows the kernel doesn't change much during training.
Allen-Zhu et al. [2019]	L	$D^{\Omega(1)}$	ReLU	$\mathcal{E}$	From 2 layers to L layers. Hard technical work. $T = D^{\Omega(1)} \log(1/\epsilon)$.
Arora et al. [2019]	2	$\frac{D^8}{\lambda_{\min}^4 \epsilon^4}$	ReLU	$\mathcal{E}_{\parallel i}$	Gets per- $\lambda_i(\mathbf{H}^\infty)$ channels.
Nguyen [2021]	L	D to D^3	ReLU	$\mathcal{E}$	No kernels. No eigenvalues. Uses algebraic geometry of ReLU (polyhedral cones). Got tight(er) width.
Song et al. [2021]	2	$\frac{D^{3/2}}{d_0}$	smooth	$\mathcal{E}$	Trains both layers from standard initialization. About $D^{3/2}$ parameters in total, up to log factors (d_0 = input dimension).
This work	L	$\Omega(D)$	any	$\mathcal{E}_{\parallel i}$ or $\mathcal{E}_d$	No kernel. Rate matrix $\mathbf{R}$. Per- α_d rates (conjectured). Ghost $\lambda_i = 0$ error. Convergence is proved only for $M \geq D$ with f_1 fixed, assuming $\alpha_d \geq \alpha_{\min}$ throughout.

Mean Field Theory: Mei et al. [2018] provided an alternative infinite-width analysis via mean field theory, capturing SGD dynamics of two-layer networks as a nonlinear PDE. Unlike the NTK/lazy regime, mean field theory permits feature learning, but still requires width tending to infinity. Our finite-width ($M \geq \Omega(D)$) analysis is complementary to both the NTK and mean field frameworks.

Removing Redundant Rows: It is well known that having extra width $M \gg D$ allows the neural network to overfit any data (at the risk of not generalizing). Separately, linear independence of the data representations $\mathbf{X}$ (requiring at least $M \geq D$) eliminates spurious valleys. However, a faulty common intuition suggests that, beyond this, overparameterization improves optimization by expanding the basins of attraction and creating additional paths to global minima. Formal versions of this idea include Venturi et al. [2018], who showed that sufficiently wide networks have no “spurious valleys” (disconnected sub-level sets), and Freeman & Bruna [2017], who proved that level sets become topologically connected at large width.

Observed: Ji and Telgarsky [2019] observed that gradient descent remains confined to the subspace spanned by the training data.

Generalizing: Early experience shows that slightly overfitting ($M = D$) has the danger of not generalizing the training data to unseen data well. More recently, however, Zhang et

al. [2017] showed that deep nets with $M \gg D$ in practice generalize fine on real data. So something beyond parameter counting is going on. A complementary line of work studies the *implicit bias* of gradient descent—why, among all solutions with zero training error, GD selects particular ones. Soudry et al. [2018] proved that for linearly separable data, gradient descent on logistic loss converges in direction to the max-margin (SVM) solution, even without explicit regularization.

Our Contribution: Once the data representation $\mathbf{X}$ has been fixed, we prove that extra width beyond $M = \text{Rank}(\mathbf{X})$ contributes nothing: the weight component $\mathbf{W}_\perp$ orthogonal to the data features of $\mathbf{X}$ has zero effect on the network output and because it has zero gradient it never changes. We prove that any redundant ($M > \text{Rank}(\mathbf{X})$) network is a “shadow” of the non-redundant ($M = \text{Rank}(\mathbf{X})$) case—same critical points, same basins, same convergence rates. This concerns the training loss only, not generalization.

Hardness of Training: Blum and Rivest [1992] proved that training even a 3-node network is NP-complete and Auer et al. [1996] showed that single-neuron loss surfaces can have exponentially many local minima. More recently, Khalife and Basu [2022] established that training ReLU networks is $W[1]$ -hard with respect to input dimension, and Froese and Hertrich [2022] extended intractability results to fixed-dimension settings. These results paint a bleak picture: training is hard *in general*.

Our Contribution: Our contribution is to consider small rank deficit $\hat{k} = D - \text{Rank}(\mathbf{X})$. We prove a $2^{\Omega(\hat{k})}$ lower bound, assuming the Exponential Time Hypothesis (ETH). We provide a $D^{O(M'\hat{k}^2)}$ dynamic-programming algorithm for finding $\min_{\mathbf{W}} \max_d \mathcal{E}_d$, showing the problem is polynomial for each fixed $\hat{k}$ and M' (the class XP); we do not know whether it is fixed-parameter tractable. The complexity of finding $\min_{\mathbf{W}} \sum_d \mathcal{E}_d$ is open. Hence, both sensitivity α and rank deficit $\hat{k}$ are “faucets” that continuously interpolate between the tractable and intractable regimes.

Lazy (NTK) vs. Feature-Learning Regimes: Chizat et al. [2019] empirically identified a spectrum from “Lazy Training” to “Feature-Learning” regimes.

Our Contribution:

Part I covers all-layer training on all the parameters $\theta = \langle f_{1,t}, \mathbf{W}_t, f_{3,t} \rangle$.

The regime for Part II is in the middle between the Lazy Training and Feature-Learning Regimes. It does not learn features as the M features of data point d in $\mathbf{x}_d := f_1(\mathbf{a}_d) \in \mathbb{R}^M$ are produced by a fixed pre-processing function f_1 . The post-processing function is also fixed. It is not Lazy because the rate matrix $\mathbf{R}_t = (\mathbf{G}_t^T \mathbf{G}_t) \odot (\mathbf{X}^T \mathbf{X})$ changes each iteration. It is not NTK because f_3 is highly nonlinear, not approximately linear. See Section 17.1.

Part I

Suboptimal Local Minima Exist

but Are Avoided

We will now summarize and contrast our error landscape results with the previous work. After this, we will formally state and prove our theorem on this topic.

- **Mostly Zero-Error ($M \geq D$):** A foundational line of work culminates with Nguyen and Hein [2017], which proves that in wide networks, with analytic activations, almost all critical points ($\nabla_{\theta}\mathcal{E}(\theta_*)=\mathbf{0}$) are global minima ($\mathcal{E}(\theta_*)=0$).
- **Some Suboptimal:** Ding et al. [2022] proved that this “almost all” is tight because for deep networks (two or more hidden layers) with smooth activations, suboptimal local minima exist regardless of width.
- **Exist but Avoided:** We prove that gradient descent never steps onto these traps of Ding’s: every suboptimal critical point lies in a measure-zero bad set $\mathcal{B}$, which gradient descent never enters (under mild conditions, with probability one over random initialization). It can still converge to such a point in the limit (Remark 2), except in the wide case (Remark 1).
- **Overparameterized Networks ($MM' \geq D$):** We relax the wide ($M \geq D$) requirement to overparameterized ($MM' \geq D$) and still get that every critical point off $\mathcal{B}$ has zero error and that gradient descent never enters $\mathcal{B}$. Whether the error then decays to zero is open.
- **“Optimal” Ghost Error ($MM' < D$):** When underparameterized and with $\mathbf{R}$ held fixed, the non-ghost error decays to zero as described above. The ghost error, however, does not decay under a given $\mathbf{R}$. Because different local minima could end with different amounts of ghost error, we could call the worst ones suboptimal. On the other hand, given the rank constraint, decaying all but k dimensions of error to zero is the best that can be done. Hence, we could call each of these minima optimal,¹⁰ in which case gradient descent avoids all sub-“optimal” critical points.
- **More Formal:** For better clarity, let’s repeat the above statements.

$$\forall \theta \notin \mathcal{B} \ (\nabla_{\theta}\mathcal{E}(\theta)=\mathbf{0}) \rightarrow (\mathcal{E}(\theta)=0)$$

$$\exists \theta \in \mathcal{B} \ (\nabla_{\theta}\mathcal{E}(\theta)=\mathbf{0}) \ \& \ (\mathcal{E}(\theta) \neq 0)$$

$$\overline{\mathcal{B}}_{\text{Thm 1.1}} := \{\theta : \text{Rank}(\mathbf{X}(\theta))=D\} \ (M \geq D)$$

$$\overline{\mathcal{B}}_{\text{NH}} := \{\theta : (\det(\mathbf{H}(\theta)) \neq 0)$$

& the zeros of the activation function’s derivatives are not being accessed

& $(\forall_{\ell \geq k_{\mathbf{X}}} \text{Rank}(\mathbf{W}_{\ell}(\theta))=\text{full})\} \ (M \geq D)$

$$\overline{\mathcal{B}}_{\text{Our}} := \{\theta : \text{Rank}(\mathbf{R}(\theta))=\min(MM', D)\} \ (\text{any } MM' \text{ and } D).$$

- **Definitions:**

- **Landscape Point:** θ is a setting of the network parameters.
- **Critical:** $\nabla_{\theta}\mathcal{E}(\theta_*)=\mathbf{0}$ means that the point is critical/flat in the error landscape making gradient descent stall.
- **Optimal:** $\mathcal{E}(\theta)=0$ (or best possible).
- **Almost All:** The bad set $\mathcal{B}$ has measure zero.
- **Data representation:** The network has at least one layer of width $M \geq D$. Let $\mathbf{x}_d := f_1(\mathbf{a}_d) \in \mathbb{R}^M$ be the values on these M nodes on data point/input $\mathbf{a}_d$. We need these to be linearly independent.
- **Hessian:** The second derivative $\mathbf{H}(\theta) := \nabla_{\theta}^2 \mathcal{E}(\theta) \in \mathbb{R}^{|\theta| \times |\theta|}$ is a complicated matrix which NH wants to have full rank (nondegenerate).

¹⁰ Animal Farm: “Some are more Equal than others.”

- **Sensitive:**

- **Flat:** We all know that if all the activation functions (sigmoid or a ReLU) are being accessed in regions with derivatives effectively zero, then the error landscape goes flat becoming one big critical point ($\nabla_{\theta}\mathcal{E}(\theta_*) \approx \mathbf{0}$) at which there is no reason for the error to be minimum.
- **Analytic Activations:** Nguyen and Hein avoid this mathematically by requiring the network’s activations to be analytic: the derivative of which has isolated zeros — a discrete (countable) set, hence measure zero. This is why they exclude ReLU. Our rank-preservation proof also uses analyticity, so it excludes ReLU too.
- **Full-Rank Downstream:** Once the activation functions are assured to have a nonzero derivative, this gradient signal must propagate from the output backward through the network to the wide layer represented by $\mathbf{x}$ without being annihilated. Nguyen and Hein ensure this by requiring that the fully connected weights $\mathbf{W}_{\ell}$ at each of these layers have full column rank.
- **Our Replacement Assumption:** In contrast, our later layers, which we call $f_3 : \mathbb{R}^{M'} \rightarrow \mathbb{R}$, can have any structure whatsoever as long as it is sensitive to its inputs: For every data point d and every iteration t of gradient descent, we require $\alpha_{dt} := \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2 \geq \alpha_{\min} > 0$.
- **Cleaner:** This captures both the weight rank and activation sensitivity in one condition. $\alpha_{dt} > 0$ means the composed Jacobian through all downstream layers is nonzero. When it fails, the cause is irrelevant: whether from a bad $\mathbf{W}_{\ell}$ or a saturated activation, either way $\alpha_{dt} = 0$.
- **Our $\overline{\mathbf{B}}_{\text{Our}}$ Set:** In our proof, we express all of these needs by the one statement $\text{Rank}(\mathbf{R}) = \min(MM', D)$.

- **Faucets from Tractable to Intractable:** This confirms our key thesis that complexity and randomness of f_1 and f_3 , sensitivity α_d , $\mathbf{R}$ rank deficit k , and $\mathbf{X}^T \mathbf{X}$ rank deficit $\hat{k}$ act as faucets that smoothly interpolate between tractable (Part II) and intractable (Part III) regimes.

Summary of Part II:

- **Rate and Jacobian Matrices:** Define $\mathbf{J}_t \in \mathbb{R}^{D \times MM'}$ to be the Jacobian of the network outputs at iteration t and $\mathbf{R}_t = \mathbf{J}_t \mathbf{J}_t^T \in \mathbb{R}^{D \times D}$ to be the rate matrix.
- **Error Decay:** The error vector at iteration t is $\mathbf{e}_t = \prod_{t' < t} (\mathbf{I} - 2\eta \mathbf{R}_{t'}) \mathbf{e}_0$ where $\mathbf{e}_t := [e_{dt}] \in \mathbb{R}^D$ is the error vector and $e_{dt} = f_{\mathbf{W}_t}(\mathbf{a}_d) - y_d^* \in \mathbb{R}$ the error for data point d . (Theorem 2.1).
- **Zero-Error Critical Points:** At any critical point with $\text{Rank}(\mathbf{R}_*) = D$ the error is zero. $\nabla_{\mathbf{W}} \mathcal{E}_* = 0 \iff \|\nabla_{\mathbf{W}} \mathcal{E}_*\|^2 = 4 \mathbf{e}_*^T \mathbf{R}_* \mathbf{e}_* = 0 \iff \mathbf{e}_* = \mathbf{0}$, since $\mathbf{R}$ is positive definite.
- **Ghost Error:** The current ghost error is the projection of the current error vector $\mathbf{e}_t$ on the null space of the current rate matrix $\mathbf{R}_t$. It is not detected by the gradient $\|\nabla_{\mathbf{W}} \mathcal{E}_t\|^2 = 4 \mathbf{e}_t^T \mathbf{R}_t \mathbf{e}_t$ and does not decay under the current $\mathbf{R}_t$.

Why $\alpha_d > 0$ matters: Gradient descent stalls when the error landscape is flat. For example, suppose the output of the network is a sigmoid and the target for a point is $y_d^* = 5$. Gradient descent pushes $\mathbf{z}_d$ toward infinity, driving the slope α_d of the sigmoid toward zero. The gradient $\nabla_{\mathbf{W}} \mathcal{E}$ approaches zero as $\mathbf{z}_d \rightarrow \infty$, but never reaches it. The error converges to $(1-5)^2$: not zero, but this is optimal given the bounded range of the sigmoid. More subtly, cancellations in f_3 can produce isolated zeros of α_d ; we conjecture that gradient descent

with discrete steps jumps over these with only a temporary drop in rate.¹¹ Real networks mitigate flatness with Adam, batch normalization, and learning-rate schedules—techniques that stabilize α_d while still allowing $\mathbf{R}$ to evolve. The condition $\alpha_{dt} \geq \alpha_{\min} > 0$ in the theorem below captures this requirement.

Losing Randomness: Random $\langle f_{1*}, \mathbf{W}_*, f_{3*} \rangle$ ensures that both the data representation $\mathbf{X}$ and the rate matrix $\mathbf{R}$ have full rank with probability one (Theorem 2.6.1 & 3). Initially all three are random. If gradient descent keeps f_1 and f_3 fixed then they stay random. This keeps $\text{Rank}(\mathbf{X}) = D$. However, as gradient descent evolves the weights from $\mathbf{W}_0$ to $\mathbf{W}_*$, $\mathbf{W}_*$ is no longer random. We therefore cannot appeal to a generic-position argument and need the more careful analysis below.

Practical motivation for the engineering bound: If f_3 is a sigmoid, then even as its input $z \rightarrow \infty$, the gradient never fully vanishes. The bound below guarantees that a small gradient implies small error.

Theorem 1 (No Suboptimal Local Minima Reached).

Suppose:

- **Sufficiently Complex:** The pre- and post-processing networks f_1 and f_3 need to include at least one fully connected layer of weights and non-linear analytic activation functions.¹²
- **Initially Random:** Initialize $\langle f_{1,0}, \mathbf{W}_0, f_{3,0} \rangle$ randomly.
- **Fix or Train:** Gradient descent can fix f_1 and f_3 and train on only $\mathbf{W}$ or it can train on all three.
- **Steep:** The sensitivity of f_3 at each data point d 's pre-activation value $\mathbf{z}_{dt}$ at iteration t needs to remain bounded: $\forall dt \alpha_{dt} := \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_{dt})\|^2 \geq \alpha_{\min} > 0$.

It follows that:

1. **No Suboptimal Local Minima** ($M \geq D$):

If f_1 is fixed randomly, then $\text{Rank}(\mathbf{X}) = D$ w.p.1. (Theorem 2.6.1)

As such, any critical point $(\nabla_{\mathbf{W}} \mathcal{E} = \mathbf{0})$ ¹³ is optimal ($\mathcal{E} = 0$).

(This is Nguyen and Hein [2017]'s trivial case and reproved here).

Moreover, with f_1 fixed, $\lambda_{\min}(\mathbf{R}_t) \geq \lambda_{\min}(\mathbf{X}^T \mathbf{X}) \alpha_{\min} > 0$ at every iteration, so the error decays to zero (for a small enough step size; Remark 1).

2. **Rank Preservation** ($MM' \geq D$): As gradient descent proceeds, $\text{Rank}(\mathbf{R}_t) = D$ for all finite iterations t with probability one.¹⁴ This alone does not prove that the error decays to zero (Remark 2).

3. **Rank Preservation** ($MM' < D$): As gradient descent proceeds, $\text{Rank}(\mathbf{R}_t) = MM'$ for all finite iterations t with probability one. Under a fixed $\mathbf{R}$, the non-ghost error decays to zero, while the $k := D - MM'$ -dimensional ghost error persists. You can decide whether to consider this “optimal” or not.^{10 15}

¹¹See Figure 5 and row “Funny Drops” in Section 12.

¹²This needs a width condition. If the first fully connected layer of f_3 has F hidden units, every gradient $\mathbf{g}_d$ lies in an F -dimensional space, so $\text{Rank}(\mathbf{R}) \leq \min(F, M') M$. Full rank therefore needs $FM \geq \min(MM', D)$.

¹³For Ding to consider it critical, he would also need $\nabla_{f_1} \mathcal{E} = \mathbf{0}$ and $\nabla_{f_3} \mathcal{E} = \mathbf{0}$. Having a smaller critical set only makes our result stronger.

¹⁴As gradient descent evolves, the rate matrix $\mathbf{R}_t$ may pass near or converge to a $\mathbf{R}_*$ of reduced rank, making the error decay rate slow.

¹⁵Even if f_1 and f_3 are fixed, different initializations of $\mathbf{W}_0$ give different network outputs and hence different initial error $e_d := f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*$. Different $\mathbf{W}_0$ have different initial $\mathbf{R}_0$ with different null spaces and

4. **Engineering Bound:** More specifically, if $\|\nabla_{\mathbf{W}}\mathcal{E}\| \leq \epsilon$,¹⁶

$$\text{then } \forall d, |e_{ng,d}| \leq \frac{\epsilon}{2} \sqrt{(\mathbf{R}^+)_{dd}} = \frac{\epsilon}{2} \sqrt{\sum_{i=1}^{D'} v_{di}^2 / \lambda_i} \approx \frac{\epsilon}{2} \sqrt{\text{Avg}_{i \leq D'} 1 / \lambda_i}.$$
¹⁷

Remark 1 (Wide case: the error decays). For positive semidefinite $\mathbf{A}$ and $\mathbf{B}$, the Schur product theorem gives $\lambda_{\min}(\mathbf{A} \odot \mathbf{B}) \geq \lambda_{\min}(\mathbf{A}) \min_d B_{dd}$, because $(\mathbf{A} - \lambda_{\min}(\mathbf{A})\mathbf{I}) \odot \mathbf{B}$ is positive semidefinite. With $\mathbf{A} = \mathbf{X}^T \mathbf{X}$ and $\mathbf{B} = \mathbf{G}_t^T \mathbf{G}_t$ (entries $\mathbf{g}_d^T \mathbf{g}_d$, diagonal entries α_{dt}), the Steep assumption gives $\lambda_{\min}(\mathbf{R}_t) \geq \mu := \lambda_{\min}(\mathbf{X}^T \mathbf{X}) \alpha_{\min}$ at every iteration. When $M \geq D$ and f_1 is fixed and random, $\mathbf{X}$ does not change and $\lambda_{\min}(\mathbf{X}^T \mathbf{X}) > 0$ with probability one. Along gradient flow, $\frac{d}{dt}\mathcal{E} \leq -\|\nabla_{\mathbf{W}}\mathcal{E}\|^2 = -4\mathbf{e}^T \mathbf{R}_t \mathbf{e} \leq -4\mu \mathcal{E}$ (training f_3 as well only adds to the decrease), so $\mathcal{E}_t \leq \mathcal{E}_0 e^{-4\mu t}$. For gradient descent $\theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{E}$, the same inequality $\|\nabla \mathcal{E}\|^2 \geq 4\mu \mathcal{E}$ (a Polyak–Łojasiewicz condition) gives $\mathcal{E}_{t+1} \leq (1 - 2\eta\mu) \mathcal{E}_t$ whenever $\eta \leq 1/L$, where L bounds the second derivative of $\mathcal{E}$ between consecutive iterates. When $M < D$, $\lambda_{\min}(\mathbf{X}^T \mathbf{X}) = 0$ and this argument gives nothing.

Remark 2 (Full rank is not enough). Let $D = 2$, $M = 1$, $M' = 2$ (so $MM' = D$), $x_1 = 1$, $x_2 = -1$, $f_3(z_1, z_2) = z_1 + (z_2^2 - 1)^2$, both targets $y_d^* = 2$, and $\mathbf{W} = (w_1, w_2)^T$. Then $\mathcal{E} = 2w_1^2 + 2((w_2^2 - 1)^2 - 2)^2$, every $\alpha_d \geq 1$, and $\det(\mathbf{R}) = 64w_2^2(w_2^2 - 1)^2$, so $\mathbf{R}$ is singular only when $w_2 \in \{0, \pm 1\}$. Run gradient descent $\mathbf{W} \leftarrow \mathbf{W} - \eta \nabla_{\mathbf{W}} \mathcal{E}$ with $0 < \eta < 1/32$, starting with $0 < |w_2| < 1$ (which has positive probability). Each step multiplies w_2 by a factor in $(0, 1)$, so $w_2 \neq 0$ and $\mathbf{R}_t$ has full rank at every iteration. Yet it converges to $w_1 = w_2 = 0$, a strict local minimum with $\mathcal{E} = 2$, where $\lambda_{\min}(\mathbf{R}) = 0$. So rank preservation and $\alpha_{\min} > 0$ do not by themselves give convergence; a lower bound on $\lambda_{\min}(\mathbf{R}_t)$ along training is needed. For $M < D \leq MM'$ this is open (Section 22).

Remark 3 (Hardness). At the other extreme, Blum and Rivest [1992] prove that training neural networks is NP-complete. This implies an exponential number of suboptimal local minima (traps). Their reduction operates precisely where our conditions fail: $M = \frac{1}{3}D$; the last layer f_3 has neither the complexity nor randomness needed; the data representation $\mathbf{X}$ has rank $D - \hat{k}$; and the gradient is effectively $\alpha_d = 0$ because the activation functions are so narrow that they look like Boolean thresholds. See Theorem 4.

7 Wide Networks ($M \geq D$)

This section restates and reproves Nguyen and Hein [2017]’s trivial case.

Theorem 1.1 (No Suboptimal Local Minima ($M \geq D$)).

If f_1 is fixed randomly, then $\text{Rank}(\mathbf{X}) = D$ w.p.1. (Theorem 2.6.1)

Any critical point ($\nabla_{\mathbf{W}}\mathcal{E} = \mathbf{0}$) is optimal with $\mathcal{E}_d = 0$ for every point for which $\alpha_d > 0$.

(This is Nguyen and Hein [2017]’s trivial case and reproven here).

hence different decomposition of this error into ghost and non-ghost. Also when $\mathbf{R}_t$ varies (which will occur differently along different gradient descent paths), the ghost subspace rotates: at each step, the non-ghost component under the current $\mathbf{R}_t$ decays. If $\mathbf{R}_t$ keeps changing, different amounts of the total error may eventually decay. (see Section 17.8)

¹⁶ The gradient $\nabla_{\mathbf{W}}\mathcal{E} = 2\mathbf{J}^T \mathbf{e}$ automatically ignores ghost error: $\mathbf{J}^T \mathbf{e}_{\text{ghost}} = \mathbf{0}$ since $\|\mathbf{J}^T \mathbf{e}_{\text{ghost}}\|^2 = \mathbf{e}_{\text{ghost}}^T \mathbf{R} \mathbf{e}_{\text{ghost}} = 0$. So “non-ghost gradient” is redundant—the gradient is already non-ghost.

¹⁷ $\mathbf{R}^+$ is the “pseudo-inverse” of $\mathbf{R}$, as it follows the eigenvector definition of inverse excluding the zero eigenvalues. When $k = 0$, $\mathbf{R}^+ = \mathbf{R}^{-1}$. Worst-case: $(\mathbf{R}^+)_{dd} \leq 1/\lambda_{\min}^+$. Budget: $\sum_d (\mathbf{R}^+)_{dd} = \sum_{i=1}^{D'} 1/\lambda_i$. Delocalized: $(\mathbf{R}^+)_{dd} \approx \text{Avg}_{i \leq D'} 1/\lambda_i$. See Theorem 2.6.8’ for bounds on the eigenvalues of $\mathbf{R}$.

Proof. Let $\mathbf{W}$ be a critical point, meaning the gradient $\nabla_{\mathbf{W}}\mathcal{E}$ vanishes. We prove that this implies the error is zero. See Figure 1.

Step 0: Fix a target training point d : We prove that the neural network gives the right answer $f_{\mathbf{W}}(\mathbf{a}_d) = y_d^*$ with zero error by constructing a weight change $\Delta\mathbf{W}_d$ that moves exactly the one output $f_{\mathbf{W}}(\mathbf{a}_d)$ while keeping all other outputs fixed.

Step 1: Isolate the Feature: Since the feature vectors $\mathbf{x}_d$ are linearly independent, there exists a dual vector $\mathbf{u}_d \in \mathbb{R}^M$ such that:

$$\mathbf{u}_d^T \mathbf{x}_d = 1 \quad \text{and} \quad \mathbf{u}_d^T \mathbf{x}_{d'} = 0 \quad (\text{for } d' \neq d)$$

(i.e., $\mathbf{X}^T \mathbf{u}_d = \mathbf{1}_d$.) This vector $\mathbf{u}_d$ acts as a filter that “picks out” only the d^{th} training point.

Step 2: Align with the Gradient: Let $\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d) \in \mathbb{R}^{M'}$ be the gradient of the final layer. Since $\alpha_d = \|\mathbf{g}_d\|^2 > 0$ by hypothesis, $\mathbf{g}_d \neq \mathbf{0}$. We define our weight update matrix $\Delta\mathbf{W}_d = \mathbf{g}_d \mathbf{u}_d^T \in \mathbb{R}^{M' \times M}$. When we apply this specific change $\Delta\mathbf{W}_d$:

- **Target point d :** Pre-activation change $\partial\mathbf{z}_d = (\Delta\mathbf{W}_d)\mathbf{x}_d = \mathbf{g}_d(\mathbf{u}_d^T \mathbf{x}_d) = \mathbf{g}_d(1)$.
The output change is $\nabla_{\mathbf{z}} f_3(\mathbf{z}_d) \cdot \partial\mathbf{z}_d = \mathbf{g}_d \cdot \mathbf{g}_d = \|\mathbf{g}_d\|^2 = \alpha_d > 0$.
- **Other points d' :** $\mathbf{u}_d^T \mathbf{x}_{d'} = 0$, so $\partial\mathbf{z}_{d'} = \mathbf{0}$. The output change is exactly 0.

Step 3: The Directional Derivative: Consider moving the weights in this direction by a scalar amount r . The new weights are $\mathbf{W}' = \mathbf{W} + r\Delta\mathbf{W}_d$. Since $\mathbf{W}$ is a critical point, the derivative of the error in any direction must be zero, and hence:

$$\left. \frac{\partial \mathcal{E}}{\partial r} \right|_{r=0} = \nabla_{\mathbf{W}} \mathcal{E} \cdot \Delta\mathbf{W}_d = 0$$

Step 4: Calculating the Change in Error: We compute this derivative explicitly by summing over all training points d' : (By Step 2, $\partial\mathbf{z}_{d'} = \mathbf{0}$ for every $d' \neq d$, so every term but $d' = d$ vanishes below; the surviving term is expanded by the chain rule.)

$$\frac{\partial \mathcal{E}}{\partial r} = \sum_{d'=1}^D \frac{\partial (f_{\mathbf{W}}(\mathbf{a}_{d'}) - y_{d'}^*)^2}{\partial r} = 2(f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*) \nabla_{\mathbf{z}} f_3(\mathbf{z}_d) \cdot \frac{\partial \mathbf{z}_d}{\partial r} = 2(f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*) \cdot \|\mathbf{g}_d\|^2$$

because all terms vanish except for d . Since $\frac{\partial \mathcal{E}}{\partial r} = 0$ and $\|\mathbf{g}_d\|^2 > 0$ by hypothesis, it must be that $f_{\mathbf{W}}(\mathbf{a}_d) = y_d^*$. This was proved independently for each training point d . It follows that every critical point is a global minimum. $\square$

8 Rank Preservation (any MM')

This section expands Theorem 1.2&3 into parts and proves each.

Theorem 1.2&3 (Rank Preservation ($\forall dt \alpha_{dt} \geq \alpha_{\min} > 0$):) *Let $D' := \min(MM', D)$. As gradient descent proceeds, $\text{Rank}(\mathbf{R}_t) = D'$ for all finite iterations t with probability one.*¹³

*Let $\theta_t = \langle f_{1,t}, \mathbf{W}_t, f_{3,t} \rangle \in \mathbb{R}^P$ denote the full parameter vector at iteration t . Gradient descent may fix f_1 and f_3 and train only $\mathbf{W}$, or train all three; the results below hold either way.*¹⁸

¹⁸When f_1 and f_3 are fixed, θ reduces to $\mathbf{W}$ and every statement below holds with $\mathbf{W}$ in place of θ .

Define $\mathcal{B} = \{\theta : \text{Rank}(\mathbf{R}(\theta)) < D'\}$ to be the set of degenerate configurations where the rate matrix loses rank below its generic value D' .

1. $\mathcal{B}$ is a measure-zero analytic subvariety of the parameter space (Theorem 2.6.3).
2. With probability one over random initialization θ_0 (for every step size η outside a countable set), gradient descent never enters $\mathcal{B}$: $\forall t \in \mathbb{N}, \theta_t \notin \mathcal{B}$ and hence $\text{Rank}(\mathbf{R}_t) = D'$.
3. If additionally $\lambda_{\min}^+(\mathbf{R}_t) \geq \mu > 0$ for all t , then the non-ghost error converges to zero (when $MM' < D$, this is proved only for a fixed $\mathbf{R}$: if $\mathbf{R}_t$ changes, its ghost subspace moves). This floor does not follow from items 1–2 together with $\alpha_{dt} \geq \alpha_{\min}$ (Remark 2). It does hold in the wide case (Remark 1).

Proof of Theorem 1.2.2. Let us begin with analytic (not piecewise-analytic) activations.

Step 1: $\mathcal{B}$ has measure zero. Stated in Theorem 1.2.1 and proved in Theorem 2.6.3.

Step 2: Each GD iterate is analytic in θ_0 . The GD map is $F(\theta) = \theta - 2\eta \nabla_{\theta} \mathcal{E}(\theta)$. When f_1 and f_3 are fixed, this reduces to $F(\mathbf{W}) = \mathbf{W} - 2\eta \nabla_{\mathbf{W}} \mathcal{E}(\mathbf{W})$. For analytic activation functions, $\mathcal{E}$ is analytic in θ , so F is analytic. By induction, $\theta_t = F^t(\theta_0)$ is analytic in θ_0 .

Step 3: The condition $\theta_t \in \mathcal{B}$ reduces to the zero of an analytic function. Note $\theta_t \in \mathcal{B}$ iff $\text{Rank}(\mathbf{R}(\theta_t)) < D'$.

- **When $D' = D$:** $\text{Rank}(\mathbf{R}) < D$ iff $\det(\mathbf{R}(\theta_t)) = 0$. Define $h_t(\theta_0, \eta) := \det(\mathbf{R}(F^t(\theta_0)))$.
- **When $D' = MM' < D$:** The $D \times D$ matrix $\mathbf{R}$ always has rank $\leq MM' < D$, so $\det(\mathbf{R}) = 0$ is useless as a detector. Instead, note $\text{Rank}(\mathbf{R}) = \text{Rank}(\mathbf{J})$ where $\mathbf{J} \in \mathbb{R}^{D \times MM'}$ is the Jacobian. The square Gram matrix $\mathbf{J}^T \mathbf{J} \in \mathbb{R}^{MM' \times MM'}$ satisfies $\det(\mathbf{J}^T \mathbf{J}) = 0$ iff $\text{Rank}(\mathbf{J}) < MM' = D'$. Define $h_t(\theta_0, \eta) := \det(\mathbf{J}(F^t(\theta_0))^T \mathbf{J}(F^t(\theta_0)))$.

In both cases, $\theta_t \in \mathcal{B}$ iff $h_t(\theta_0, \eta) = 0$, and h_t is analytic in θ_0 (composition of analytic functions).

Step 4: $h_t(\theta_0, \eta)$ is not identically zero. At $\eta = 0$ we have $F^t(\theta_0) = \theta_0$, so $h_t(\theta_0, 0) = h(\theta_0) \neq 0$ (generic θ_0 gives $\text{Rank}(\mathbf{R}) = D'$). Since h_t is analytic jointly in (θ_0, η) and not identically zero at $\eta = 0$, it remains not identically zero for all but countably many $\eta > 0$: fix θ° with $h(\theta^\circ) \neq 0$; then $\eta \mapsto h_t(\theta^\circ, \eta)$ is analytic and non-zero at $\eta = 0$, so its zeros are isolated. Over all t this excludes only countably many step sizes, and we assume η is not one of them. (Some step sizes must be excluded: for $\mathcal{E} = w^2$, the map $F(w) = (1 - 4\eta)w$ sends every w to the single point 0 when $\eta = \frac{1}{4}$.)

Step 5: The set $\mathcal{B}_t := \{\theta_0 : h_t(\theta_0, \eta) = 0\}$ has measure zero. By Step 4, $h_t(\cdot, \eta)$ is a non-identically-zero analytic function of θ_0 . The zero set of any non-identically-zero real-analytic function has measure zero.

Step 6: $\Pr[\theta_t \in \mathcal{B}] = 0$ for each finite t . By Step 3, $\Pr[\theta_t \in \mathcal{B}]$ is the probability mass of the set $\mathcal{B}_t$, which by Step 5 has measure zero.

Step 7: Countable union. $\Pr[\exists t \in \mathbb{N} : \theta_t \in \mathcal{B}] \leq \sum_{t=0}^{\infty} \Pr[\theta_t \in \mathcal{B}] = 0$. With probability one over random θ_0 , gradient descent never enters $\mathcal{B}$ at any finite iteration.

Extension to piecewise-analytic activations. [Withdrawn. This extension fails when $M < D$. With ReLU, f_3 is linear on each region of its input space, so all $\mathbf{z}_d$ in one region share one gradient $\mathbf{g}$. If $\mathbf{W}$ is near $\mathbf{0}$ and $\mathbf{0}$ lies inside a region (true for generic biases), then all $\mathbf{z}_d$ lie in that region and $\mathbf{R} = \|\mathbf{g}\|^2 \mathbf{X}^T \mathbf{X}$ has rank at most M , which is less than $D' = \min(MM', D)$ when $M < D$ and $M' \geq 2$. These $\mathbf{W}$ form an open set, so $\mathcal{B}$ does not have empty interior, and a random initialization lands in it with positive probability.] $\square$

Proof of Theorem 1.2.3 ($MM' \geq D$). Suppose θ_0 is chosen randomly and gradient descent proceeds as $\theta_0, \theta_1, \theta_2, \dots, \theta_t, \dots$. By Theorem 1.2.2, with probability one, $\forall t \ \theta_t \notin \mathcal{B}$ and $\text{Rank}(\mathbf{R}_t) = D$. This means that all the eigenvalues $\lambda_{it} > 0$ of $\mathbf{R}_t$ are positive and hence the error converges each iteration t at some rate $r_t > 0$ giving $e_{dt}^2 \leq \prod_{t' < t} (1 - 4\eta r_{t'}) \mathcal{E}_0 \approx e^{-4\eta \sum_{t' < t} r_{t'}} \mathcal{E}_0$. The problem is that in the sigmoid-output-target-5 example, $\mathbf{z}_t$ grows and r_t converges quickly to zero, so the error converges to something nonzero. Requiring $\alpha_{dt} \geq \alpha_{\min} > 0$ rules out this example, but not the one in Remark 2, where r_t still tends to zero. We therefore assume the floor directly: if $r_t \geq \lambda_{\min}(\mathbf{R}_t) \geq \mu > 0$ for all t , the bound above gives $e_{dt}^2 \leq e^{-4\eta \mu t} \mathcal{E}_0 \rightarrow 0$. (Theorem 2.4, cited here before, holds only under a coupling hypothesis; see Section 14.) $\square$

Proof of Theorem 1.2.3 ($MM' < D$). By Theorem 1.2.2, with probability one, $\forall t \ \text{Rank}(\mathbf{R}_t) = MM' = D - k$. This means that k of its eigenvalues $\lambda_{it} = 0$ correspond to k ghost channels whose error does not decay under the current $\mathbf{R}_t$. The non-ghost error, however, decays as described previously. Its rate of decay rate is at least the smallest non-zero eigenvalue, which by assumption is $r_t \geq \lambda_{\min}^+(\mathbf{R}_t) \geq \mu > 0$. The conjectured value of $\lambda_{\min}^+$ (Theorem 2.6.8) is for a random $\mathbf{R}$, not for $\mathbf{R}_t$ along training. $\square$

9 Engineering/Quantified Bounds (Robustness)

This section restates and proves Theorem 1.4.

Theorem 1.4 (Engineering Bound). *If $\text{Rank}(\mathbf{R}) = D'$ and $\|\nabla_{\mathbf{W}} \mathcal{E}\| \leq \epsilon$,¹⁶ then $\forall d, |e_{\text{ng},d}| \leq \frac{\epsilon}{2} \sqrt{(\mathbf{R}^+)_{dd}} = \frac{\epsilon}{2} \sqrt{\sum_{i=1}^{D'} v_{di}^2 / \lambda_i} \approx \frac{\epsilon}{2} \sqrt{\text{Avg}_{i \leq D'} 1 / \lambda_i}$.¹⁷*

Proof. Step 1: Define the Jacobian dual. Since $\text{Rank}(\mathbf{R}) = D'$ and $\mathbf{R} = \mathbf{J}\mathbf{J}^T$, the Jacobian $\mathbf{J} \in \mathbb{R}^{D \times MM'}$ has rank D' . Define $\mathbf{w}_d \in \mathbb{R}^{MM'}$ to be the minimum-norm vector satisfying $\mathbf{J}\mathbf{w}_d = Q\mathbf{1}_d$, where $Q = \mathbf{R}\mathbf{R}^+$ projects onto the range of $\mathbf{R}$ (i.e., $(\mathbf{J}\mathbf{w}_d)_{d'} = (Q\mathbf{1}_d)_{d'}$).

Step 2: Extract the individual error.

$$\begin{aligned}
2|e_{\text{ng},d}| &= 2|(Q\mathbf{1}_d)^T \mathbf{e}| && (e_{\text{ng},d} = (Q\mathbf{e})_d; Q \text{ symmetric}) \\
&= 2|(\mathbf{J}\mathbf{w}_d)^T \mathbf{e}| && (\mathbf{J}\mathbf{w}_d = Q\mathbf{1}_d) \\
&= 2|\mathbf{w}_d^T \mathbf{J}^T \mathbf{e}| && (\text{Transpose}) \\
&= |\mathbf{w}_d^T \nabla_{\mathbf{W}} \mathcal{E}| && (\nabla_{\mathbf{W}} \mathcal{E} = 2\mathbf{J}^T \mathbf{e}) \\
&\leq \|\mathbf{w}_d\| \cdot \|\nabla_{\mathbf{W}} \mathcal{E}\| && (\text{Cauchy-Schwarz}) \\
&\leq \epsilon \|\mathbf{w}_d\| && (\text{Hypothesis})
\end{aligned}$$

Step 3: Compute $\|\mathbf{w}_d\|^2$. The minimum-norm solution is $\mathbf{w}_d = \mathbf{J}^T \mathbf{R}^+ \mathbf{1}_d$.

$$\begin{aligned}
\|\mathbf{w}_d\|^2 &= \mathbf{1}_d^T \mathbf{R}^+ \mathbf{J}\mathbf{J}^T \mathbf{R}^+ \mathbf{1}_d && (\text{Expand}) \\
&= \mathbf{1}_d^T \mathbf{R}^+ \mathbf{R} \mathbf{R}^+ \mathbf{1}_d && (\mathbf{J}\mathbf{J}^T = \mathbf{R}; \mathbf{R}^+ \mathbf{R} \mathbf{R}^+ = \mathbf{R}^+) \\
&= \mathbf{1}_d^T \mathbf{R}^+ \mathbf{1}_d = (\mathbf{R}^+)_{dd}
\end{aligned}$$

Hence $|e_{\text{ng},d}| \leq \frac{\epsilon}{2} \sqrt{(\mathbf{R}^+)_{dd}}$.

Recovery of wide-case bound: When $M \geq D$ and $\text{Rank}(\mathbf{X}) = D$, we can verify that $\mathbf{w}_d = \text{vec}(\mathbf{g}_d \mathbf{u}_d^T) / \alpha_d$ satisfies $\mathbf{J} \mathbf{w}_d = \mathbf{1}_d$ (by the computation in Step 2 of the proof in Section 7). Then $\|\mathbf{w}_d\|^2 = \|\mathbf{g}_d\|^2 \|\mathbf{u}_d\|^2 / \alpha_d^2 = u_d^2 / \alpha_d$, recovering $|e_d| \leq \frac{\epsilon u_d}{2\sqrt{\alpha_d}}$. $\square$

Lengths of $\mathbf{u}_d$

Claim 1. *There are vectors $\mathbf{u}_d$ with the required properties $\mathbf{X}^T \mathbf{u}_d = \mathbf{1}_d$ such that $\sum_d \|\mathbf{u}_d\|^2 = \sum_i \frac{1}{\hat{\lambda}_i}$; $\|\mathbf{u}_d\|^2 \leq 1/\hat{\lambda}_{\min}$; for random f_1 , $\|\mathbf{u}_d\|^2 \approx \text{Avg}_i \frac{1}{\hat{\lambda}_i}$.*

Proof.

- **Intuition:** Though they have different rotations, $\{\mathbf{u}_d\}$ and $\{\hat{\mathbf{u}}_i\}$ are both bases for $\text{col}(\mathbf{X})$ (only $\{\hat{\mathbf{u}}_i\}$ is orthonormal). The more these bases are aligned, the more properties they inherit.

- **Eigenvalues:** The thin SVD is $\mathbf{X}^T = \hat{\mathbf{V}} \hat{\mathbf{\Sigma}} \hat{\mathbf{U}}^T$, where the columns $\hat{\mathbf{v}}_i \in \mathbb{R}^D$ of $\hat{\mathbf{V}}$ and $\hat{\mathbf{u}}_i \in \mathbb{R}^M$ of $\hat{\mathbf{U}}$ are orthonormal, and $\hat{\mathbf{\Sigma}} = \text{diag}(\sqrt{\hat{\lambda}_i})$ with $\hat{\lambda}_i$ being the eigenvalues of $\mathbf{X}^T \mathbf{X}$. For each $\hat{\mathbf{u}}_i$, $\mathbf{X}^T \hat{\mathbf{u}}_i = \hat{\mathbf{V}} \hat{\mathbf{\Sigma}} (\hat{\mathbf{U}}^T \hat{\mathbf{u}}_i) = \hat{\mathbf{V}} \hat{\mathbf{\Sigma}} \mathbf{1}_i = \sqrt{\hat{\lambda}_i} \hat{\mathbf{v}}_i$ and so $\frac{\|\mathbf{X}^T \hat{\mathbf{u}}_i\|^2}{\|\hat{\mathbf{u}}_i\|^2} = \hat{\lambda}_i$.

- **Length:** If these vectors lined up so that $\frac{\|\mathbf{X}^T \mathbf{u}_d\|^2}{\|\mathbf{u}_d\|^2} = \hat{\lambda}_i$, then because $\|\mathbf{X}^T \mathbf{u}_d\|^2 = \|\mathbf{1}_d\|^2 = 1$, we would have $\|\mathbf{u}_d\|^2 = \frac{1}{\hat{\lambda}_i}$.

- **Minimum:** Not more than one $\mathbf{u}_d$ can line up with $\hat{\mathbf{u}}_{\min}$.

- **Solution:** The minimum-norm solution to $\mathbf{X}^T \mathbf{u}_d = \mathbf{1}_d$ is $\mathbf{u}_d = \mathbf{X}(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{1}_d$.

As required $\mathbf{X}^T \mathbf{u}_d = \mathbf{X}^T [\mathbf{X}(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{1}_d] = (\mathbf{X}^T \mathbf{X})(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{1}_d = \mathbf{1}_d$.

- **Eigen Expansion** $\|\mathbf{u}_d\|^2 = \sum_i \frac{\hat{v}_{di}^2}{\hat{\lambda}_i}$:

Proof: Write the eigendecomposition $\mathbf{X}^T \mathbf{X} = \hat{\mathbf{V}} \hat{\mathbf{\Lambda}} \hat{\mathbf{V}}^T = \sum_i \hat{\lambda}_i \hat{\mathbf{v}}_i \hat{\mathbf{v}}_i^T$, so $(\mathbf{X}^T \mathbf{X})^{-1} = \sum_i \frac{1}{\hat{\lambda}_i} \hat{\mathbf{v}}_i \hat{\mathbf{v}}_i^T$.

Then $(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{1}_d = \sum_i \frac{\hat{v}_{di}}{\hat{\lambda}_i} \hat{\mathbf{v}}_i$, where $\hat{v}_{di} = \hat{\mathbf{v}}_i^T \mathbf{1}_d$ is the d^{th} entry of $\hat{\mathbf{v}}_i$. Hence $\mathbf{u}_d = \mathbf{X}(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{1}_d = \sum_i \frac{\hat{v}_{di}}{\hat{\lambda}_i} \mathbf{X} \hat{\mathbf{v}}_i$. $\mathbf{X} \hat{\mathbf{v}}_i = [\hat{\mathbf{U}} \hat{\mathbf{\Sigma}} \hat{\mathbf{V}}^T] \hat{\mathbf{v}}_i = \sqrt{\hat{\lambda}_i} \hat{\mathbf{u}}_i$, giving $\mathbf{u}_d = \sum_i \frac{\hat{v}_{di}}{\sqrt{\hat{\lambda}_i}} \hat{\mathbf{u}}_i$. Since the

$\hat{\mathbf{u}}_i$ are orthonormal, $\|\mathbf{u}_d\|^2 = \sum_i \frac{\hat{v}_{di}^2}{\hat{\lambda}_i}$, as claimed.

- $\|\mathbf{u}_d\|^2 \leq \frac{1}{\hat{\lambda}_{\min}}$: $\sum_i \frac{\hat{v}_{di}^2}{\hat{\lambda}_i}$ is maximized subject to $\sum_i \hat{v}_{di}^2 = 1$ by putting all the weight on $\frac{1}{\hat{\lambda}_{\min}}$.

- $\sum_d \|\mathbf{u}_d\|^2 = \sum_d \sum_i \frac{\hat{v}_{di}^2}{\hat{\lambda}_i} = \sum_i \frac{1}{\hat{\lambda}_i} [\sum_d \hat{v}_{di}^2] = \sum_i \frac{1}{\hat{\lambda}_i} [1]$.

- For random f_1 , $\hat{v}_{di}^2 \approx 1/D$ and hence $\|\mathbf{u}_d\|^2 = \sum_i \frac{\hat{v}_{di}^2}{\hat{\lambda}_i} \approx \text{Avg}_i \frac{1}{\hat{\lambda}_i}$.

(Theorem 2.6.7 using $\mathbf{q} = [\frac{1}{\hat{\lambda}_i}]$)

$\square$

Part II

Per-Point Convergence

Part I has already established that gradient descent keeps the rate matrix at full rank (with probability one), but not that the error goes to zero (Remark 2). We now ask *how fast*—characterizing per data point decay rates.

Model Review: The pre and post-processing functions f_1 and f_3 (pre-learned or random) are fixed and assumed to have properties that occur with probability one when their parameters are chosen randomly. Gradient descent is assumed to only train on the one layer $\mathbf{W}$ so that the rate matrix $\mathbf{R}$ is easier to define. f_1 and f_3 may contain complex structures like transformers; the training of these is beyond the scope of this paper. If they contain multiple fully connected layers $\mathbf{W}_\ell$, then the total rate matrix when training them all is the sum $\mathbf{R} = \sum_\ell \mathbf{R}_\ell$; see Section 5. A lot changes, however. As $\mathbf{W}_t$ changes with iterations t , point d 's pre-activation $\mathbf{z}_{dt} := \mathbf{W}_t \mathbf{x}_d$ changes, which changes the gradient $\mathbf{g}_{dt} := \nabla_{\mathbf{z}} f_3(\mathbf{z}_{dt})$, the rate matrix $\mathbf{R}_t := (\mathbf{G}_t^T \mathbf{G}_t) \odot (\mathbf{X}^T \mathbf{X})$ and its eigenvalues λ_{it} , vectors $\mathbf{v}_{it}$, and sensitivities $\alpha_{dt} := \|\mathbf{g}_{dt}\|^2$. Often we drop the index t assumed to be the current iteration. Sometimes to simplify the math we write $(1-4\eta\alpha_d)^t$, which assumes α_d does not change with iterations. One can either take $\alpha_d = \min_t \alpha_{dt}$ for a worst-case bound, or try to interpolate for the time-varying case. Other times, we include the more general form $\mathcal{E}_{dt} = \prod_{t'=1}^t (1-4\eta\alpha_{dt'}) \mathcal{E}_{d0} \approx e^{-4\eta \sum_{t'=1}^t \alpha_{dt'}} \mathcal{E}_{d0}$. Part I already holds when all parameters are trained and continually changed. See Section 17.1 for where this regime sits between lazy training and feature learning.

The results are classified into:

- **Spectral Results:**

- **Rate Matrix:** Errors decay with $\mathbf{e}_t = \prod_t (\mathbf{I} - 2\eta \mathbf{R}_t) \mathbf{e}_0$, where $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$.
- **Spectral Channels:** The error and its decay can be decomposed into spectral channels according to the eigendecomposition of the rate matrix $\mathbf{R}$. Each channel operates at its own rate λ_i .
- **Ghost Errors:** Channels with $\lambda_i = 0$ are called *ghost* and do not decay under a given $\mathbf{R}$, or interact with others.
- **Point-wise Error:** Ultimately what we care about is point d 's error $e_d = f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*$. At iteration t , it has decayed to $e_{dt} = \sum_i (1-2\eta\lambda_i)^t v_{di} e_{i0}$, completing the non-ghost error in at most $O(\lambda_{\min}^{+1} \log(\mathcal{E}_0/\epsilon))$ iterations (for a fixed $\mathbf{R}$). (Section 11)

- **Random Network Structures:** This work required lots of research on results about random matrices in order to capture the properties arising from randomness in f_1 and f_3 . (Section 16)

- **Conjectures/Simulations:** With this randomness, the expected value of $\lambda_{\min}^+$ is conjectured and confirmed in simulations to be approximately $\alpha_{\min}$ ($MM' \gg D$) or $\frac{D}{MM'} \bar{\alpha}$ ($MM' \ll D$); near $MM' = D$ it is much smaller, about $(\sqrt{D/MM'} - 1)^2 \bar{\alpha}$ (Theorem 2.6.8).¹⁹

- **Signal vs Noise Results:** In order to learn more about the decay of the point-wise error $\mathcal{E}_d = (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2$, we partition its decay update $\Delta e_d \propto \mathbf{R} \mathbf{e} = \alpha_d e_d \pm \xi_d$ into signal $\alpha_d e_d$ (diagonal of $\mathbf{R}$) and noise ξ_d (off-diagonals) and prove whp that:

- **Negligible Noise:** When $M \geq \Omega(D)$ and the errors are comparable in size, $\mathcal{E}_d$ decays at rate α_d . (Section 13)
- **Managed Noise:** Assuming a coupling hypothesis on the noise (not yet derived from the random-network model), the weighted error $\bar{\alpha} \mathcal{E}$ decays at rate at least $\frac{1}{2} \alpha_{\text{coll}} \geq \frac{1}{2} \alpha_{\min}$. (Section 14)
- **Random Walks:** A random-walk argument initially looked promising (signal $\alpha_d e_d$ is steady; noise ξ_d has random sign). This implies a random walk where the signal

¹⁹The first arises from a 1-1 mapping from α_d to λ_{i_d} and the second because MM' active channels share the full rate budget $\sum_d \alpha_d$.

grows with T and the noise with $\sqrt{T}$. We prove that even with adversarial dependence between iterations this decay occurs with reasonable probability, provided the noise ξ_d has no bias toward the sign of the error e_d . The method fails because the sign of the noise ξ_d correlates with the sign of the error e_d . This effect arises from the ghost mechanism. (Section 15)

- **Simulations:** Randomly rotating $\mathbf{R}$ moves ghost error into decaying channels, but does not speed up the rate, and actual gradient descent struggles because either $\mathbf{R}$ refuses to rotate or $\lambda_{\min}^+$ drops too close to zero. (Section 12)

Theorem 2 (Gradient Descent Decay).

Gradient descent initially decays the non-ghost component of each data point d 's error $\mathcal{E}_d = (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2$ independently at a rate of α_d (in expectation, for errors with no preferred direction).²⁰ Depending on the setting, it either continues at that rate or slows down to $\alpha_{\min}$ or $\lambda_{\min}^+$ (conjectured; the items below say what is proved).

The number of iterations needed is between $\Theta(\alpha_d^{-1} \log(\mathcal{E}_0/\epsilon))$ and $O(\lambda_{\min}^{+,-1} \log(\mathcal{E}_0/\epsilon))$; only the upper bound is proved, and only for a fixed $\mathbf{R}$.

Spectral Results: Results are based on the spectral decomposition of the rate matrix $\mathbf{R}$.

1. **Spectral Decomposition:** (Section 11)

(1) **Overview:**

GD Decay:	$\Delta \mathbf{e} = 2\eta \mathbf{R} \mathbf{e} \in \mathbb{R}^D$	$\mathbf{e}_t = \prod_{t'=1}^t (\mathbf{I} - 2\eta \mathbf{R}_{t'}) \mathbf{e}_0$ ($\mathbf{R}$ changes with t)
Diagonal Only:	$\Delta \mathcal{E}_d = 4\eta \alpha_d \mathcal{E}_d$	$\mathcal{E}_{dt} = \prod_{t'=1}^t (1 - 4\eta \alpha_{dt'}) \mathcal{E}_{d0} \approx e^{-4\eta \sum_{t'=1}^t \alpha_{dt'}} \mathcal{E}_{d0}$
# Iterations:	$T_d = \Theta\left(\frac{1}{\eta \alpha_d} \log \frac{\mathcal{E}_{d0}}{\epsilon}\right)$	($\mathcal{E}_d$ and α_d change with t)
Where:		
Point Error:	$e_d = f_{\mathbf{W}}(\mathbf{a}_d) - y_d^* \in \mathbb{R}$	$\mathbf{e} = [e_d] \in \mathbb{R}^D$
Squared	$\mathcal{E}_d = e_d^2$	$\mathcal{E} = \sum_d \mathcal{E}_d = \ \mathbf{e}\ ^2$
Data Represent.:	$\mathbf{x}_d = f_1(\mathbf{a}_d)$	$\ \mathbf{x}_d\ = 1$ (Assume normalized)
Point Sensitivity:	$\alpha_d = \ \mathbf{g}_d\ ^2$	$\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d)$
Rate Matrix:	$r_{dd'} = (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'}) \in \mathbb{R}$	$\mathbf{R} = [r_{dd'}] = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X}) \in \mathbb{R}^{D \times D}$

(2) **Spectral Rate Matrix $\mathbf{R}$:** The eigenvectors $\mathbf{v}_i$ of the rate matrix $\mathbf{R} \in \mathbb{R}^{D \times D}$ decompose the total error $\mathcal{E} = \sum_i \mathcal{E}_{\|i}$ into spectral errors $\mathcal{E}_{\|i} = (\mathbf{v}_i^T \mathbf{e})^2 \in \mathbb{R}$, which decay at their own eigenvalue rate $r_i = \lambda_i$.

(3) **Decay over Iterations:** (Assuming $\mathbf{R}$, λ_i and $\mathbf{v}_i$ do not change).

The error for point d at iteration t has decayed to

$$e_{dt} = \sum_i (1 - 2\eta \lambda_i)^t v_{di} e_{\|i0}.$$

$$\mathbb{E}[\mathcal{E}_{dt}] = \sigma^2 \sum_i (1 - 2\eta \lambda_i)^{2t} v_{di}^2.$$

The second result requires independence between channel errors, namely, $\mathbb{E}[e_{\|i}^2] = \sigma^2$ and $\mathbb{E}[e_{\|i} e_{\|i'}] = 0$.

(4) **Rank of $\mathbf{R}$:** Let D' denote the rank of $\mathbf{R}$.

When the output network f_3 is linear, $\text{Rank}(\mathbf{R}) = \text{Rank}(\mathbf{X}) = \min(D, M)$. Random non-linearity of f_3 , however, allows its rank to be $D' := \text{Rank}(\mathbf{R}) = \min(D, MM')$.

²⁰When α_{dt} varies with t , replace $(1 - 4\eta \alpha_d)^t$ with $\prod_{t'} (1 - 4\eta \alpha_{dt'})$; taking $\alpha_d = \min_t \alpha_{dt}$ gives a worst-case bound.

(5) **Ghost Error:** Define the spectral error $\mathcal{E}_{\parallel i}$ to be ghost if its rate of decay is $\lambda_i = 0$. Gradient descent does not change it at all, until the rate matrix $\mathbf{R}$ changes. The number of these is denoted $k := D - \text{Rank}(\mathbf{R})$.

(6) **Non-Ghost Point-wise Error:** Define $\hat{\mathcal{E}}_d := (\sum_{i=1}^{D'} \sqrt{\mathcal{E}_{\parallel i}} v_{di})^2$ to be the amount of the point-wise error $\mathcal{E}_d$ that is non-ghost.

(7) **Rate of Decay:**

- **Defⁿ of Rate:** The rate of decay of point d 's error at iteration t is defined to be $\text{rate}_{dt}^{\text{all}} := 1 - \frac{\mathbb{E}[\mathcal{E}_{d(t+1)}]}{\mathbb{E}[\mathcal{E}_{dt}]}$ and non-ghost error rate $\text{rate}_{dt}^{\text{non-ghost}} := 1 - \frac{\mathbb{E}[\hat{\mathcal{E}}_{d(t+1)}]}{\mathbb{E}[\hat{\mathcal{E}}_{dt}]}$.²¹

- **Initial Rate:** $\text{rate}_{d0}^{\text{all}} = 4\eta\alpha_d$ and $\text{rate}_{d0}^{\text{non-ghost}} = 4\eta\frac{\alpha_d}{P_{dd}}$,

where $P_{dd} = \frac{\sum_{i=1}^{D'} v_{di}^2}{\sum_{i=1}^D v_{di}^2}$ is the fraction of error that is non-ghost.²²

- **Final Rate:** $\lim_{t \rightarrow \infty} \text{rate}_{dt}^{\text{all}} = 0$ and $\lim_{t \rightarrow \infty} \text{rate}_{dt}^{\text{non-ghost}} = 4\eta\lambda_{\min}^+$ (The former since ghost error does not decay – assuming there is some).

Conjectured: $\lambda_{\min}^+ \approx \alpha_{\min}$ ($MM' \gg D$) and $\lambda_{\min}^+ \approx \frac{D}{MM'} \bar{\alpha}$ ($MM' \ll D$).

- **Fraction of Time Fast:** The fast rate stalls when the error in the fast channels completes, this, however, may happen quite quickly. For example, suppose the fast channels make up 90% of the error. Removing it decreases the error by a factor of $\frac{1}{10}$ but only decreases $\log_{10}(\text{error})$ by an additive 1. In contrast, decreasing the error from 1 to $\epsilon = 10^{-6}$, requires decreasing $\log_{10}(\text{error})$ by another 6.

(8) **Extra Facts:**

- $[\mathcal{E}_{\parallel i} = 0] \rightarrow [\mathbf{e} \text{ perpendicular to } \mathbf{v}_i]$.
- $\mathcal{E} = \sum_d \mathcal{E}_d = \sum_d e_d^2 = \sum_i \mathcal{E}_{\parallel i}$
- $\mathbb{E}[\mathcal{E}_{\parallel i}] = \frac{1}{D} \mathcal{E}$
- $\mathbb{E}[\hat{\mathcal{E}}_d] \approx \frac{D'}{D} \mathcal{E}_d$ (randomly rotated $\mathbf{v}_i$)
- $\text{Avg}_i \lambda_i = \frac{1}{D} \text{Trace}(\mathbf{R}) = \text{Avg}_d \alpha_d$
- No Effect of Ghost on Non-Ghost
- If for each i , $\partial \mathcal{E}_{\parallel i} = r \mathcal{E}_{\parallel i}$, then for each d , $\partial \mathcal{E}_d = r \mathcal{E}_d$.²³

2. **Simulations:** We run simulations on random $\mathbf{R}$ to understand what we could not prove about gradient descent. The insights not mirrored in the theoretical results are as follows:

- (1) $\alpha_d \Leftrightarrow \lambda_{i_d}$: When heavily overparameterized ($MM' \gg D$) each point d has a dedicated spectral channel i_d with $\lambda_{i_d} \approx \alpha_d$. This allows each point to decay at its own rate α_d .
- (2) **Rotating $\mathbf{R}$:** We try randomly rotating $\mathbf{R}$ in order to move our current error into faster channels. As hoped, when $MM' < D$, this did manage to decay the ghost error. However, this did not keep the rate up near α_d or $\bar{\alpha}$ as hoped. The rate settled at $\lambda_{\min}^+$. (Section 12)
- (3) **Gradient Descent:** When running actual gradient descent in this same $MM' <$

²¹Depending on how neat we are feeling, we either include the 4η or not.

²² Here $\mathbf{P} = \sum_{i=1}^{D'} \mathbf{v}_i \mathbf{v}_i^T$ and being orthonormal $\sum_{i=1}^D v_{di}^2 = 1$

²³An Embarrassing Bug. One might hope that if all spectral errors $\mathcal{E}_{\parallel i}$ decay at a rate of at least r , then all point errors $\mathcal{E}_d$ also decay at a rate of at least r . However, Theorem 2.7.2 shows that a frozen point with $\alpha_d = 0$ has decay rate zero—not r . The fix is that strict equality is needed.

D range, we did not manage to get rid of the ghost error because either our rate matrices $\mathbf{R}$ refused to rotate, or $\lambda_{\min}^+$ dropped too close to zero. (Section 12)

Signal vs Noise Results: The remaining results partition the decay progress into signal (diagonal of $\mathbf{R}$) and noise (off-diagonals).

$$\begin{aligned} \text{GD Decay:} \quad \partial \mathbf{e} &= 2\eta \mathbf{R} \mathbf{e} \in \mathbb{R}^D \\ \partial e_d &= 2\eta \left(\underbrace{\alpha_d e_d}_{\text{signal } r_d e_d} + \underbrace{\sum_{d' \neq d} r_{dd'} e_{d'}}_{\text{noise } \xi_d} \right) \end{aligned}$$

Source of Randomness:

- **Random f_1 and $\mathbf{X}$:** These results use randomness in f_1 to ensure whp that the data representation $\mathbf{x}_d = f_1(\mathbf{a}_d)$ has randomly rotated eigenvectors $\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}} \mathbf{H}$. For example, the input could be a single real number $\mathbf{a}_d$, with f_1 consisting of one layer of activations followed by random weights.
- **Arbitrary f_3 :** Here we assume the worst case f_3 . This occurs when f_3 is linear giving identical gradients $\mathbf{g}_d = \mathbf{g}_{d'}$ and hence maximizing the expression $|\mathbf{g}_d^T \mathbf{g}_{d'}| = \alpha$ that appears in the noise.
- **Random f_3 :** When f_3 is chosen randomly, the gradients are approximately pairwise orthogonal, decreasing $\mathbf{g}_d^T \mathbf{g}_{d'}$ and hence the noise.

Rates of Decay: The following 2×2 table of results shows that even if the noise always subtracts, the decay occurs as stated.

	Arbitrary f_3	Random f_3
Negligible Noise: $r_d = \alpha_d$	$M \geq \Omega(D^2)^{23}$	$M \geq \Omega(D)$
Managed Noise: $r_d \in \{\alpha_d, \alpha_{\text{coll}}\}$	$M \geq \Omega(D)$	$MM' \geq \Omega(D)$

3. **Negligible Noise:** When the decay noise is small enough, it leaves only the point-wise decay rate $r_d = \alpha_d$. (This is about the current iteration; the width needed grows with $e_{\max}/e_{\min}$.) (Section 13)
4. **Managed Noise:** (Conditional on the coupling hypothesis of Section 14. Only the collective rate below is proved under it; the two-phase story is not.) Initially, all points decay at their own rate α_d .
When the fast ones are too far ahead, the cross-point noise drags everyone to the collective rate $\alpha_{\text{coll}} = \frac{\sum_d \alpha_d^2 \mathcal{E}_d}{\sum_d \alpha_d \mathcal{E}_d}$ (average of α_d weighted by $\alpha_d \mathcal{E}_d$)^{24 25} (Section 14)
5. **Random Walks:** Stochastic gradient descent is an effective way to add randomness in order to get around obstacles. We studied how this might help us handle our noise. (Section 15)

²³ Whether the Ω in these results really should be $\tilde{\Omega}$ containing an extra $\log D$ factor or not depends on the scope of the guarantee you want to obtain from the probabilistic union bound. For example, if you want to choose your favorite data point d and bound only its error decay, then Ω does the trick. On the other hand, if you want this same bound for all data points simultaneously, then you need to increase the Ω by the extra $\log D$ factor.

²⁴ Unfortunately, points with smaller α_d , decaying more slowly, will likely have larger errors $\mathcal{E}_d$ and hence contribute more to the collective weighted average. For sure $\alpha_{\text{coll}} \geq \alpha_{\min}$.

²⁵ This is one case where interpolating what happens when α_{dt} varies with time is complex.

- (1) **Random Walk:** At each iteration t , the noise has zero mean $\mathbb{E}[\xi_{dt}] = 0$ (and symmetric), giving it a random $\pm$ sign. This implies a random walk where the signal grows with T and the noise with $\sqrt{T}$.
 - (2) **Probability of Success:** The sign of the noise ξ_{dt} , though random at each iteration, likely has huge dependence across iterations. For example, if nothing changes then the noise does not change $\xi_{dt} = \xi_{d(t-1)}$. This does mean, however, with probability $\frac{1}{2}$ all the noise is helpful. Hence, you only need to run the gradient descent twice to get the results you want. We generalize this idea to handle more complex adversarial dependencies.
 - (3) **Adversarial Dependence:** The problem is that when the fast channels have decayed their error, the error that remains is in the ghost/slow channels. The way the decay $\partial e_d \propto \mathbf{R}e = \alpha_d e_d \pm \xi_d$ is supposed to work is as follows. When the pre-squared error e_d is positive, ∂e_d is also positive and is subtracted from the error. When e_d is negative, ∂e_d is negative and hence subtracting it still makes progress. However, for a ghost channel ($\lambda_i = 0$), if the error vector is $\mathbf{e} = [e_d] = \mathbf{v}_i$ then $\partial e_d \propto \mathbf{R}e = \lambda_i \mathbf{e} = \mathbf{0}$. From a random walk perspective, the sign of the noise ξ is such that when e_d and ∂e_d are positive, ξ_d is negative perfectly canceling out the signal's progress and when e_d and ∂e_d are negative, ξ_d is positive again perfectly canceling progress.
6. **Random Network Structures:** Suppose f_1 and f_3 are sufficiently complex and have randomly initialized weights, producing data representations $\mathbf{x}_d = f_1(\mathbf{a}_d)$ and activation gradients $\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d)$.
- For example, f_1 could map an input consisting of a single real number $\mathbf{a}_d$ to one layer of activations followed a fully connected layer $\mathbf{W}' \in \mathbb{R}^{M \times M}$ of random weights. The following structural properties hold with high probability.
- Items marked **Conjecture** are supported by simulations but not yet proved.
- (1) **Rank of $\mathbf{X}$:** With sigmoids, $\mathbf{X} \in \mathbb{R}^{M \times D}$ has full rank $\min(D, M)$.
With ReLU, columns may coincide (see item 5).
 - (2) **Rank of $\mathbf{G}$:** For fixed random f_3 , the set of pairwise-distinct $(\mathbf{z}_1, \dots, \mathbf{z}_D)$ for which $\mathbf{G}$ has rank less than $\min(D, M')$ is a measure-zero analytic subvariety of $\mathbb{R}^{M' \times D}$.
 - (3) **Rank of $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$:** For fixed random f_3 , the set of pairwise-distinct $(\mathbf{z}_1, \dots, \mathbf{z}_D)$ for which $\mathbf{R} \in \mathbb{R}^{D \times D}$ (surprisingly) has rank less than $\min(D, MM')$ is a measure-zero analytic subvariety of $\mathbb{R}^{M' \times D}$.
 - (4) **Approximately Normal $\mathbf{X}$ and $\mathbf{G}$:**
Each entry is $x_{md} \approx \mathcal{N}(0, \frac{1}{M})$ and $g_{(m'd)} \approx \mathcal{N}(0, \frac{1}{M'})$ (columns may be correlated).
 - (5) **Coupon Collector** (Sigmoid or ReLU activation functions and $M \geq \Omega(D \log D)$):
Each consecutive input a_d vs a_{d+1} is split by at least one neuron. $\mathbf{X}$ has full rank D . Columns are approximately independent.
 - (6) **Near-Orthonormal:** $\mathbf{X}^T \mathbf{X} \approx \mathbf{I}$ with $\mathbf{x}_d^T \mathbf{x}_{d'} = O\left(\sqrt{\frac{\log D}{M}}\right)$ for $d \neq d'$.
 - (7) **Randomly Rotated Eigenvectors:** $\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}} \mathbf{H}$ (any M vs D):
Eigenvectors of $\mathbf{X}^T \mathbf{X}$ satisfy $\hat{v}_{di} \approx \text{Norm}(0, \frac{1}{D})$.
We would love to plug in $\pm \frac{1}{D}$ for $\hat{v}_{di}$, but likely some of them will be as big as $\pm \frac{\log D}{D}$, which we do not want plugged in for all of them. Instead:

Fix $s \leq D^{\Theta(1)}$ vectors $\mathbf{q} \in \mathbb{R}^D$ depending on $\hat{\mathbf{V}}$ only through entries not being summed. Whp the following $4sDT$ sums are true simultaneously.

- $\forall d \sum_i \hat{v}_{di}^2 q_i = (1 \pm \epsilon) \frac{1}{D} \sum_i q_i$
- $\forall d |\sum_i \hat{v}_{di} q_i| \leq (1 + \epsilon) \sqrt{\frac{1}{D} \sum_i q_i^2}$
- Again with the role of d and i switched
- Again for each iteration $t \in [T]$.

Remark: In contrast, the eigenvectors of $\mathbf{R}$ are localized.

(8) **Eigenvalues (Marchenko–Pastur):**

$\mathbf{X}^T \mathbf{X}$ Proven: Eigenvalues of $\mathbf{X}^T \mathbf{X}$ satisfy $\text{CV}(\hat{\lambda}) := \frac{SDV(\hat{\lambda})}{\hat{\lambda}_{\text{avg}}} = \sqrt{\frac{D}{M}}$.

$\mathbf{R}$ Proven:

- $\bar{\lambda} = \bar{\alpha}$ and $\bar{\lambda}^+ = \frac{D}{D'} \bar{\alpha}$ (fewer channels share the trace budget $D\bar{\alpha}$).
- $D' = \min(D, MM')$ and $k = D - D'$ zero eigenvalues (ghost channels) follows from item 3.

$\mathbf{R}$ Conjectures: Verified to 1.5% across all tested values.

- $\text{CV}(\lambda) \approx \sqrt{D/MM'}$: Simulations confirm this for the CV of all D eigenvalues (including zeros). The CV of only the D' non-zero eigenvalues is different.
- $MM' \gg D$: There is a 1-1 mapping between data points and spectral channels with $\lambda_{i_d} \approx \alpha_d$. In particular, $\lambda_{\min} \approx \alpha_{\min}$. See Figure 2.A.
- $MM' \approx D$: $\lambda_{\min}^+ \approx (\sqrt{D/MM'} - 1)^2 \bar{\alpha}$

This dips dangerously close to zero, stalling decay of “non-ghost” errors.

This can be “fixed”, however, by just calling such near-zero channels ghost.

- $MM' \ll D$: No matter how diverse the data point sensitivities α_d are, the non-zero eigenvalues of $\mathbf{R}$ are tightly concentrated around $\bar{\lambda}^+ = \frac{D}{D'} \bar{\alpha}$. Hence, the minimum non-zero eigenvalue is $\lambda_{\min}^+ \approx c \frac{D}{MM'} \bar{\alpha}$ for $c \geq \frac{1}{2}$ and 1 in the limit. See Figure 2.B.

7. **Extensions and Generalizations:** The theorem shows when simplifying assumptions can be relaxed.

- (1) **Equal $\hat{\lambda}_i$:** If $\forall i \hat{\lambda}_i = \hat{\lambda}$, then each data point d ’s error $\mathcal{E}_d$ decays at its own rate $r_d = \alpha_d \hat{\lambda}$.
- (2) **Ignoring Stalled Points:** Points whose sensitivity α_d is too small can be ignored. Their error will not degrade quickly but they do not disturb the other points.
- (3) **Worst case Weights $\mathbf{W}$:** Not a problem.
- (4) **Second-Order Effects:** The second order effects in gradient descent can be bounded if the step size η is sufficiently small.
- (5) **Generalizing to $\langle \alpha, \beta \rangle$ -PL Sensitivity:** An easy adjustment in the proof.
- (6) **Removing Redundant Rows:** A common intuition suggests that overparameterization improves optimization by expanding the basins of attraction and creating additional paths to global minima. We prove that this is not the case by proving that any redundant ($M > \hat{D}$) network is a “shadow” of the non-redundant ($M = \hat{D}$) case—same critical points, same basins, same convergence rates.
- (7) **Ghost Error Decays:** Simple models of rate-matrix rotation and stochastic gradient descent show that the ghost error decays to zero.

10 Gradient Descent

This section proves Theorem 2.1.1. The rest of this section explains the proof for the novice reader. This part can be skipped by any expert reader who already follows the proof.

Theorem 2.1.1 (Gradient Descent Decay).

Gradient descent decays each data point d 's error $\mathcal{E}_d$ at iteration t as follows

It is exact to first order in η and holds for every nonlinear f_3 .

Rate Matrix: $r_{dd'} = (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'}) \in \mathbb{R}$

GD Decay: $\Delta \mathbf{e} = 2\eta \mathbf{R} \mathbf{e} \in \mathbb{R}^D$

Signal: $\Delta \mathcal{E}_d = 4\eta \alpha_d \mathcal{E}_d$

Proof.

$$\begin{aligned}
\Delta \mathbf{W} &= \eta \nabla_{\mathbf{W}} \mathcal{E} && \text{(Step of Gradient descent)} \\
&= \eta (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{X}^T && (\mathbf{Z} = \mathbf{W} \mathbf{X} \text{ Matrix chain rule}) \\
&= \eta \sum_d (\nabla_{\mathbf{z}_d} \mathcal{E}_d) \mathbf{x}_d^T && \text{(point-wise)} \\
\nabla_{\mathbf{z}_d} \mathcal{E}_d &= \nabla_{\mathbf{z}_d} e_d^2 = 2(\nabla_{\mathbf{z}_d} e_d) e_d && \text{(Chain Rule)} \\
&= 2(\nabla_{\mathbf{z}_d} (f_3(\mathbf{z}_d) - y_d^*)) e_d && \text{(Definition)} \\
&= 2(\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)) e_d && \text{(Derivative)} \\
&= 2\mathbf{g}_d e_d && \text{(Definition of } \mathbf{g}_d) \\
\Delta \mathbf{W} &= 2\eta \sum_d e_d \mathbf{g}_d \mathbf{x}_d^T && \text{(Combining)} \\
\Delta e_d &\approx \mathbf{g}_d^T \Delta \mathbf{z}_d && \text{(first order)} \\
&= \mathbf{g}_d^T \Delta \mathbf{W} \mathbf{x}_d && (\Delta \mathbf{z}_d = \Delta \mathbf{W} \mathbf{x}_d) \\
&= \mathbf{g}_d^T \left(2\eta \sum_{d'} e_{d'} \mathbf{g}_{d'} \mathbf{x}_{d'}^T \right) \mathbf{x}_d && \text{(Combining)} \\
&= 2\eta \sum_{d'} (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_{d'}^T \mathbf{x}_d) e_{d'} && \text{(Rearranging inner products)} \\
&= 2\eta \sum_{d'} r_{dd'} e_{d'} && \text{(Definition of } r_{dd'}) \\
\Delta \mathbf{e} &= 2\eta \mathbf{R} \mathbf{e} && \text{(Matrix Form)} \\
\mathbf{e}(t+1) &= \mathbf{e}(t) - \Delta \mathbf{e}(t) = (\mathbf{I} - 2\eta \mathbf{R}) \mathbf{e}(t) && \text{(Recurrence)} \\
\mathbf{e}(t) &= (\mathbf{I} - 2\eta \mathbf{R})^t \mathbf{e}_0 && \text{(Decaying over iterations)} \\
\mathcal{E}_d &:= e_d^2 && \text{(Squared Error)} \\
\frac{d\mathcal{E}_d}{dt} &= 2e_d \frac{de_d}{dt} && \text{(Calculus)} \\
\Delta \mathcal{E}_d &= 2e_d \Delta e_d && \text{(Discrete)} \quad \square
\end{aligned}$$

We now explain this proof in more detail by analyzing one step of gradient descent. Given our training data $[\langle \mathbf{a}_d, y_d^* \rangle \mid d \in [D]]$ and the current value of our weights $\mathbf{W} \in \mathbb{R}^{M' \times M}$ (and the weights in f_1 and f_3), we do forward propagation to learn the outputs $f_{\mathbf{W}}(\mathbf{a}_d) = f_3(\mathbf{W} \cdot f_1(\mathbf{a}_d))$ of this network on this data and the error $\mathcal{E} = \sum_d (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2 \in \mathbb{R}$. Then we do backward propagation to compute the derivatives and update the weights $\mathbf{W}$ in the way that takes a step in the direction of steepest descent. This paper does not consider changing f_1 or f_3 , though this would be handled in the same way. Finally, we determine how this change in weights changes the error.

Forward Propagation: We group values into vectors and matrices.

Post- f_1 Representation $\mathbf{x}_d$ and $\mathbf{X}$: Let $\mathbf{x}_d = f_1(\mathbf{a}_d) \in \mathbb{R}^M$ denote this representation for data point d and let $\mathbf{X} = [\mathbf{x}_d \mid d \in [D]] \in \mathbb{R}^{M \times D}$ package these into the vector of such vectors (matrix). These are fixed throughout gradient descent on weights $\mathbf{W}$.

Weights $\mathbf{W}$: Let $\mathbf{W} \in \mathbb{R}^{M' \times M}$ denote the current values of the weights.

Pre- f_3 Representation $\mathbf{z}_d$ and $\mathbf{Z}$: Let $\mathbf{z}_d = \mathbf{W}\mathbf{x}_d \in \mathbb{R}^{M'}$ and $\mathbf{Z} = [\mathbf{z}_d \mid d \in [D]] = \mathbf{W}\mathbf{X} \in \mathbb{R}^{M' \times D}$ denote these current representations.

Readout Function f_3 : Let $f_3(\mathbf{z}_d) \in \mathbb{R}$ denote the current output of the neural network for this data point.

Errors e_d , $\mathbf{e}$, $\mathcal{E}_d$, and $\mathcal{E}$: Let $e_d = (f_3(\mathbf{z}_d) - y_d^*) \in \mathbb{R}$ denote the pre-squared error and $\mathcal{E}_d = e_d^2$ the squared error for this point. Let $\mathbf{e} = [e_d \mid d \in [D]] \in \mathbb{R}^D$ denote the vector of these values and $\mathcal{E} = \sum_d \mathcal{E}_d = \|\mathbf{e}\|^2$ denote the total error.

Back Propagation vs Forward Change: We will casually switch between the infinitesimal single value derivatives $\frac{\partial \mathcal{E}}{\partial w_{MM'}}$, the discrete changes $\frac{\Delta \mathcal{E}}{\Delta w_{MM'}}$ made in this iteration of gradient descent, and the same using matrix notation.

Gradient $\nabla_{\mathbf{W}} \mathcal{E} \in \mathbb{R}^{M' \times M}$: This is a matrix with entries $\frac{\partial \mathcal{E}}{\partial w_{MM'}}$ specifying how the single valued global error $\mathcal{E}$ changes with each of the $M' \times M$ weights in $\mathbf{W}$. It is also the vector pointing in the direction of the steepest ascent within the weight space.

Gradient Descent: This is the direction taken $\mathbf{W}_{t+1} = \mathbf{W}_t - \eta \nabla_{\mathbf{W}} \mathcal{E} \in \mathbb{R}^{M' \times M}$.

Change in Weights: $\Delta \mathbf{W} := \eta \nabla_{\mathbf{W}} \mathcal{E} \in \mathbb{R}^{M' \times M}$ is the matrix of entries $\Delta w_{MM'}$ giving how much the weight **decreases**²⁶ in a step of gradient descent.

Change in $\mathbf{z}_d$ gives Change in Error: We now explain the following two lines:

$$\begin{aligned} \nabla_{\mathbf{z}_d} \mathcal{E} &= \nabla_{\mathbf{z}_d} \mathcal{E}_d = \nabla_{\mathbf{z}_d} e_d^2 = 2(\nabla_{\mathbf{z}_d} e_d) e_d \\ \Delta e_d &\approx \mathbf{g}_d^T \Delta \mathbf{z}_d \text{ (first order).} \end{aligned}$$

Changing $\mathbf{z}_d$ for one Point: Here $\mathbf{z}_d$ denotes the pre- f_3 representation for the one data point d . Changing it changes the output for this point but not that for the other points. It follows that $\nabla_{\mathbf{z}_d} \mathcal{E} = \nabla_{\mathbf{z}_d} \mathcal{E}_d$.

Change Δe_d : For each data point $d \in [D]$, Δe_d is the amount (positive or negative) that this point's pre-squared error $e_d = (f_3(\mathbf{z}_d) - y_d^*) \in \mathbb{R}$ changes in this iteration of gradient descent.

Changing One Entry of $\mathbf{z}_d \in \mathbb{R}^{M'}$: For each feature $m' \in [M']$, there is a node entering the $f_3(\mathbf{z}_d)$ part of the network and $\mathbf{z}_{m'd}$ is its value on d . $\Delta \mathbf{z}_{m'd}$ is how much it changes in this iteration of gradient descent.

Derivative: The gradient $\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d) \in \mathbb{R}^{M'}$ is a vector of the derivatives $\frac{\partial f_3(\mathbf{z}_d)}{\partial \mathbf{z}_{m'd}}$ giving how this change affects this point's output.

Chain Rule: $\Delta e_d \approx \frac{\partial e_d}{\partial f_3(\mathbf{z}_d)} \frac{\partial f_3(\mathbf{z}_d)}{\partial \mathbf{z}_{m'd}} \Delta \mathbf{z}_{m'd}$ gives how much e_d changes because of this change in $\mathbf{z}_{m'd}$. Note that because $e_d = (f_3(\mathbf{z}_d) - y_d^*)$, we have that $\frac{\partial e_d}{\partial f_3(\mathbf{z}_d)} = 1$.

Changing All of $\mathbf{z}_d$: Then because all of these $\mathbf{z}_{m'd}$ change in parallel by gradient descent, we get the total change in error by summing these $\Delta e_d \approx \sum_{m'} 1 \frac{\partial f_3(\mathbf{z}_d)}{\partial \mathbf{z}_{m'd}} \Delta \mathbf{z}_{m'd}$ which is the dot product $\Delta e_d \approx \mathbf{g}_d^T \Delta \mathbf{z}_d$.

First Order: Because this step of gradient descent is not infinitesimal, there are second order (curvature) effects too which we are ignoring here. See Section 17.5 for a discussion about the second order changes.

Change $\mathbf{W}$ gives Change in $\mathbf{z}_d$: We now explain the following line:

$$\nabla_{\mathbf{W}} \mathcal{E} = [\nabla_{\mathbf{Z}} \mathcal{E}] \mathbf{X}^T$$

We can hand wave this using notational magic. $\mathbf{Z} = \mathbf{W}\mathbf{X}$ and hence $\frac{\partial \mathbf{Z}}{\partial \mathbf{W}} \stackrel{“}{=} \mathbf{X}$ even though

²⁶The actual update is $\mathbf{W} \leftarrow \mathbf{W} - \Delta \mathbf{W}$. This avoids a huge number of negative signs in the paper. For example, $\Delta \mathcal{E}_{\parallel i} = r \mathcal{E}_{\parallel i}$ means exponential decay.

they have different dimensions. Then $\frac{\partial \mathcal{E}}{\partial \mathbf{W}} = \frac{\partial \mathcal{E}}{\partial \mathbf{Z}} \frac{\partial \mathbf{Z}}{\partial \mathbf{W}}$. Converting to gradients gives $\nabla_{\mathbf{W}} \mathcal{E} = [\nabla_{\mathbf{Z}} \mathcal{E}] \frac{\partial \mathbf{Z}}{\partial \mathbf{W}} = [\nabla_{\mathbf{Z}} \mathcal{E}] \mathbf{X}^T$. The only other thing to check is that the matrix dimensions in the multiplication match up. $\nabla_{\mathbf{W}} \mathcal{E} = [\nabla_{\mathbf{Z}} \mathcal{E}] \mathbf{X}^T$ has dimensions $(M' \times M) = (M' \times D)(D \times M)$ as needed.

The rest of the proof is straightforward.

11 Spectral Decomposition of the Error

This section restates and proves Theorem 2.1.2-8. These decompose the error and its decay into spectral channels according to the eigendecomposition of the rate matrix $\mathbf{R}$. Each channel operates at its own rate λ_i . Channels with $\lambda_i = 0$ are called *ghost* and do not decay under a given $\mathbf{R}$. We show that the initial rate of decay of point d 's error is α_d and it eventually settles at $\lambda_{\min}^+$ whose expected value is conjectured and confirmed in simulations to be approximately $\alpha_{\min} (MM' \gg D)$ or $\frac{D}{MM'} \bar{\alpha} (MM' \ll D)$; near $MM' = D$ it is much smaller, about $(\sqrt{D/MM'} - 1)^2 \bar{\alpha}$ (Theorem 2.6.8). When underparameterized, non-ghost errors decay rapidly. Ghost error does not decay under fixed $\mathbf{R}$.

Theorem 2.1.2 (Spectral Rate Matrix $\mathbf{R}$).

The eigenvectors $\mathbf{v}_i$ of the rate matrix $\mathbf{R} \in \mathbb{R}^{D \times D}$ decompose the total error $\mathcal{E} = \sum_i \mathcal{E}_{\parallel i}$ into spectral errors $\mathcal{E}_{\parallel i} = (\mathbf{v}_i^T \mathbf{e})^2 \in \mathbb{R}$, which decay at their own eigenvalue rate $r_i = \lambda_i$.

Proof:

- **Open Problem:** This rate matrix $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$ is complicated and we know of no results about its eigenvectors $\mathbf{v}_i$ and eigenvalues λ_i . This may be an open problem.
- **Spectral Errors:** One can do a change in basis and decompose the vector of errors $\mathbf{e} = [(f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*) \mid d \in [D]] \in \mathbb{R}^D$ into $\mathbf{e} = \sum_i \sqrt{\mathcal{E}_{\parallel i}} \mathbf{v}_i \in \mathbb{R}^D$, where the i^{th} spectral error $\mathcal{E}_{\parallel i} = (\mathbf{v}_i^T \mathbf{e})^2 \in \mathbb{R}$ is defined to be the square of the component of the error in the $\mathbf{v}_i$ direction.²⁷
- **Decay Rate:** We prove that the decay rate of $\mathcal{E}_{\parallel i}$ is λ_i as follows:

$$\begin{aligned}
\mathcal{E}_{\parallel i t} &= (\mathbf{v}_i^T \mathbf{e}_t)^2 && \text{(Definition)} \\
&= (\mathbf{v}_i^T (\mathbf{I} - 2\eta \mathbf{R}) \mathbf{e}_{t-1})^2 && \text{(Using } \mathbf{e}_t = (\mathbf{I} - 2\eta \mathbf{R}) \mathbf{e}_{t-1} \text{)} \\
&= ((\mathbf{v}_i^T - 2\eta \mathbf{v}_i^T \mathbf{R}) \mathbf{e}_{t-1})^2 && \text{(Rearranged)} \\
&= ((\mathbf{v}_i^T - 2\eta \lambda_i \mathbf{v}_i^T) \mathbf{e}_{t-1})^2 && (\mathbf{v}_i^T \mathbf{R} = \lambda_i \mathbf{v}_i^T \text{ since } \mathbf{R} \text{ symmetric)} \\
&= (1 - 2\eta \lambda_i)^2 (\mathbf{v}_i^T \mathbf{e}_{t-1})^2 && \text{(Pull out constant)} \\
&= (1 - 2\eta \lambda_i)^2 \mathcal{E}_{\parallel i t-1} && \text{(Definition)} \\
&\approx (1 - 4\eta \lambda_i) \mathcal{E}_{\parallel i t-1} && \text{(Drop the } (2\eta \lambda_i)^2 \text{ term assumed to be small)} \\
&\approx e^{-4\eta \lambda_i t} \mathcal{E}_{\parallel i 0} \\
\text{rate}_i &= 1 - \frac{\mathcal{E}_{\parallel i t}}{\mathcal{E}_{\parallel i t-1}} && \text{(Definition)} \\
&= 4\eta \lambda_i && \text{(For neatness we drop the constant } 4\eta \text{ from the rate)}
\end{aligned}$$

²⁷Here $\sqrt{\mathcal{E}_{\parallel i}} = \mathbf{v}_i^T \mathbf{e} \in \mathbb{R}$ is the length of the component/projection of the error $\mathbf{e}$ in the $\mathbf{v}_i$ direction. Hence, $\sqrt{\mathcal{E}_{\parallel i}} \mathbf{v}_i = \mathbf{v}_i \sqrt{\mathcal{E}_{\parallel i}} = \mathbf{v}_i \mathbf{v}_i^T \mathbf{e} \in \mathbb{R}^D$ is this component/projection. Summing up these perpendicular components gets us back the original vector $\sum_i \sqrt{\mathcal{E}_{\parallel i}} \mathbf{v}_i = \sum_i \mathbf{v}_i (\mathbf{v}_i^T \mathbf{e}) = (\sum_i \mathbf{v}_i \mathbf{v}_i^T) \mathbf{e} = (\mathbf{I}) \mathbf{e} \in \mathbb{R}^D$.

□

Theorem 2.1.3 (Decay over Iterations). (Assuming $\mathbf{R}$, λ_i and $\mathbf{v}_i$ do not change).

The error for point d at iteration t has decayed to

$$e_{dt} = \sum_i (1 - 2\eta\lambda_i)^t v_{di} e_{\parallel i0}.$$

$$\mathbb{E}[\mathcal{E}_{dt}] = \sigma^2 \sum_i (1 - 2\eta\lambda_i)^{2t} v_{di}^2.$$

The second result requires independence between channel errors,

namely, $\mathbb{E}[e_{\parallel i}^2] = \sigma^2$ and $\mathbb{E}[e_{\parallel i} e_{\parallel i'}] = 0$.

Proof:

- **Spectral Decomposition:** Theorem 2.1.2 decomposes the error vector $\mathbf{e} = \sum_i \sqrt{\mathcal{E}_{\parallel i}} \mathbf{v}_i \in \mathbb{R}^D$ into spectral channels within which the error decays at its own eigenvalue rate λ_i .
- **A Quantum Reality:**²⁸ Reality = $\sum_i e_{\parallel i0} \text{Universe}_i$
Imagine we are living in a *reality* that is in the *superposition* of many alternative universes, one for each spectral channel $i \in [D]$, each weighted by time-zero spectral error $\sqrt{\mathcal{E}_{\parallel i}} = e_{\parallel i} = \mathbf{v}_i^T \mathbf{e}$. For time $t=0$, this is also denoted $e_{\parallel i0}$.
- **Universe/Channel i :** Suppose we lived in a universe/channel (indexed by i) in which the pre-squared point error $e_d := f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*$ just happens to be v_{di} so that the error vector $\mathbf{e}$ is the eigenvector $\mathbf{v}_i$.
- **Decay is Easy:** Theorem 2.1.1 proves that $\Delta \mathbf{e} = 2\eta \mathbf{R} \mathbf{e} \in \mathbb{R}^D$. Being an eigenvector, this becomes $\Delta \mathbf{e} = 2\eta \mathbf{R} \mathbf{e} = 2\eta \mathbf{R} \mathbf{v}_i = 2\eta \lambda_i \mathbf{v}_i$. After one iteration, the error vector becomes $\mathbf{e} - \Delta \mathbf{e} = \mathbf{v}_i - 2\eta \lambda_i \mathbf{v}_i = (1 - 2\eta \lambda_i) \mathbf{v}_i$ and after t iterations it becomes $(1 - 2\eta \lambda_i)^t \mathbf{v}_i$.
- **A Single Reality:** We are now ready to recombine the universes into a single reality using the equation “Reality = $\sum_i e_{\parallel i0} \text{Universe}_i$.” In Universe_i , the error vector $\mathbf{e}_t$ at iteration t is $(1 - 2\eta \lambda_i)^t \mathbf{v}_i$. Combining these gives the reality that $\mathbf{e}_t = \sum_i e_{\parallel i0} [(1 - 2\eta \lambda_i)^t \mathbf{v}_i]$.
- **Pre-Square Error e_{dt} of Point d at Iteration t :**
The result $e_{dt} = \sum_i (1 - 2\eta \lambda_i)^t v_{di} e_{\parallel i0}$ follows by picking out the d^{th} component.
- **Squared Error $\mathcal{E}_{dt}$ of Point d at Iteration t :**
 - **Claim:** If there is independence between channel errors,
 $\mathbb{E}[e_{\parallel i}^2] = \sigma^2$ and $\mathbb{E}[e_{\parallel i} e_{\parallel i'}] = 0$,
then $\mathbb{E}[\mathcal{E}_{dt}] = \sigma^2 \sum_i (1 - 2\eta \lambda_i)^{2t} v_{di}^2 + 0$.
 - **Proof:**

$$\begin{aligned}
\mathbb{E}[\mathcal{E}_{dt}] &= \mathbb{E}[e_{dt}^2] && \text{(Definition)} \\
&= \mathbb{E}\left[\left(\sum_i e_{\parallel i} (1 - 2\eta \lambda_i)^t v_{di}\right)^2\right] && \text{(Previous claim)} \\
&= \sum_i \mathbb{E}[e_{\parallel i}^2] (1 - 2\eta \lambda_i)^{2t} v_{di}^2 + \sum_{i \neq i'} \mathbb{E}[e_{\parallel i} e_{\parallel i'}] \dots && \text{(Expand square, linearity of } \mathbb{E}) \\
&= \sigma^2 \sum_i (1 - 2\eta \lambda_i)^{2t} v_{di}^2 + 0 && \text{(Independence)}
\end{aligned}$$

□

Theorem 2.1.4 (Rank of $\mathbf{R}$). Let D' denote the rank of $\mathbf{R}$.

Let $k := D - D'$ denote the rank deficit and hence the number of ghost channels.

When the output network f_3 is linear, $\text{Rank}(\mathbf{R}) = \text{Rank}(\mathbf{X}) = \min(D, M)$.

Random Non-linearity of f_3 , however, allows its rank to be $D' := \text{Rank}(\mathbf{R}) = \min(D, MM')$.

Proof:

²⁸The analogy with quantum mechanics is informal — no complex amplitudes or measurement collapse — but the algebraic structure of superposition and squaring is genuinely parallel.

- **Linear f_3 :** With f_3 linear, $\mathbf{g}_d = \mathbf{g} \in \mathbb{R}^{M'}$ is the same for all d , so $\alpha_d = \|\mathbf{g}\|^2 = \alpha$ and we can simplify $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X}) = \alpha \mathbf{X}^T \mathbf{X}$. With probability one this has rank $\min(D, M)$ (Theorem 2.6.3)
- **Non-linear f_3 :** $\mathbf{R} \in \mathbb{R}^{D \times D}$ has the surprising rank $\min(D, MM')$, because it is the Hadamard product of two Gram matrices. (Theorem 2.6.3)
Note that with nonlinear f_3 , ghost channels vanish once $MM' \geq D$, which is much earlier than the $M \geq D$ required for linear networks.

□

Theorem 2.1.5 (Ghost Error).

Define the spectral error $\mathcal{E}_{\parallel i}$ to be *ghost* if its rate of decay is $\lambda_i = 0$.

Gradient descent does not change it at all, until the rate matrix $\mathbf{R}$ changes. The number of these is denoted $k := D - D'$.

Proof. Item 2 gives that the rate of a channel is λ_i , which for ghost channels is zero. Item 3 gives that $e_{dt} = \sum_i (1 - 2\eta\lambda_i)^t v_{di} e_{\parallel i0}$. Note how for the ghost terms $(1 - 2\eta\lambda_i)^t = 1$. □

Theorem 2.1.6 (Non-Ghost Point-wise Error).

Define $\hat{\mathcal{E}}_d := \left(\sum_{i=1}^{D'} \sqrt{\mathcal{E}_{\parallel i}} v_{di} \right)^2$ to be the amount of the point-wise error $\mathcal{E}_d$ that is non-ghost.

“Proof”:

- As said, the change of basis decomposes the error vector into $\mathbf{e} = \sum_{i=1}^{D'} \sqrt{\mathcal{E}_{\parallel i}} \mathbf{v}_i + \sum_{i'=D'+1}^D \sqrt{\mathcal{E}_{\parallel i'}} \mathbf{v}_{i'}$, where the ghost errors are indexed by i' .
Define $\hat{\mathbf{e}} = \sum_{i=1}^{D'} \sqrt{\mathcal{E}_{\parallel i}} \mathbf{v}_i = \mathbf{P} \mathbf{e} \in \mathbb{R}^D$ to be the non-ghost components of the error, where $\mathbf{P} = \sum_{i=1}^{D'} \mathbf{v}_i \mathbf{v}_i^T$ projects onto the non-ghost eigenvectors. Note that $\hat{\mathbf{e}}$ is perpendicular to all ghost eigenvectors $\mathbf{v}_{i'}$.
- For point d , define the amount of the point-wise non-ghost error to be $\hat{e}_d = [\hat{\mathbf{e}}]_d \in \mathbb{R}$ and $\hat{\mathcal{E}}_d = \hat{e}_d^2 \in \mathbb{R}$.²⁹

□

Theorem 2.1.7 (Rate of Decay).

- **Defⁿ of Rate:** The rate of decay of point d 's error at iteration t is defined to be $\text{rate}_{dt}^{\text{all}} := 1 - \frac{\mathbb{E}[\mathcal{E}_{d(t+1)}]}{\mathbb{E}[\mathcal{E}_{dt}]}$ and non-ghost error $\text{rate}_{dt}^{\text{non-ghost}} := 1 - \frac{\mathbb{E}[\hat{\mathcal{E}}_{d(t+1)}]}{\mathbb{E}[\hat{\mathcal{E}}_{dt}]}$.²¹
- **Initial Rate:** $\text{rate}_{d0}^{\text{all}} = 4\eta\alpha_d$ and $\text{rate}_{d0}^{\text{non-ghost}} = 4\eta \frac{\alpha_d}{P_{dd}}$,
where $P_{dd} = \frac{\sum_{i=1}^{D'} v_{di}^2}{\sum_{i=1}^D v_{di}^2}$ is the fraction of error that is non-ghost.²²
- **Final Rate:** $\lim_{t \rightarrow \infty} \text{rate}_{dt}^{\text{all}} = 0$ and $\lim_{t \rightarrow \infty} \text{rate}_{dt}^{\text{non-ghost}} = 4\eta\lambda_{\min}^+$
(The former is zero since ghost error does not decay – assuming there is some).
Conjectured: $\lambda_{\min}^+ \approx \alpha_{\min}$ ($MM' \gg D$) and $\lambda_{\min}^+ \approx \frac{D}{MM'} \bar{\alpha}$ ($MM' \ll D$).
- **Fraction of Time Fast:** The fast rate stalls when the error in the fast channels completes; this, however, may happen quite quickly. For example, suppose the fast channels make up 90% of the error. Removing it decreases the error by a factor of $\frac{1}{10}$ but only decreases $\log_{10}(\text{error})$ by an additive 1. In contrast, decreasing the error from 1 to $\epsilon = 10^{-6}$, requires decreasing $\log_{10}(\text{error})$ by another 6.

²⁹Note that $\mathbf{v}_i$ and hence this vector are still in the point-wise basis.

Proof:

$$\begin{aligned}
\mathbf{R} &= (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X}) && \text{(Definition)} \\
r_{dd} &:= (\mathbf{g}_d^T \mathbf{g}_d) (\mathbf{x}_d^T \mathbf{x}_d) = \alpha_d 1 && \text{(Singling out } d^{th} \text{ diagonal)} \\
\mathbf{R} &= \mathbf{V} \mathbf{\Lambda} \mathbf{V}^T && \text{(Eigen decomposition)} \\
r_{dd} &= \sum_{i=1}^D \lambda_i v_{di}^2 = \alpha_d && \text{(Singling out } d^{th} \text{ diagonal, transitive, *)} \\
\text{rate}_{d0}^{\text{all}} &= 1 - \frac{\mathbb{E}[\mathcal{E}_{d(0+1)}]}{\mathbb{E}[\mathcal{E}_{d0}]} && \text{(Definition)} \\
&= 1 - \frac{\sum_{i=1}^D (1-2\eta\lambda_i)^{2(0+1)} v_{di}^2}{\sum_{i=1}^D (1-2\eta\lambda_i)^{2(0)} v_{di}^2} && \text{(Theorem 2.1.3)} \\
&\approx 1 - \frac{\sum_{i=1}^D (1-4\eta\lambda_i) v_{di}^2}{\sum_{i=1}^D v_{di}^2} && \text{(Remove } 4\eta^2 \lambda_i^2) \\
&= 1 - \frac{\sum_{i=1}^D v_{di}^2}{\sum_{i=1}^D v_{di}^2} + 4\eta \frac{\sum_{i=1}^D \lambda_i v_{di}^2}{\sum_{i=1}^D v_{di}^2} && \text{(Rearrange)} \\
&= 1 - \frac{1}{1} + 4\eta \frac{\alpha_d}{1} = 4\eta \alpha_d && \text{(Orthonormal & *)} \\
P_{dd} &= \frac{\sum_{i=1}^D v_{di}^2}{\sum_{i=1}^D v_{di}^2} = \frac{\sum_{i=1}^{D'} v_{di}^2}{1} && \text{(Fraction of error that is non-ghost)} \\
\text{rate}_{d0}^{\text{non-ghost}} &= 4\eta \frac{\sum_{i=1}^{D'} \lambda_i v_{di}^2}{\sum_{i=1}^{D'} v_{di}^2} && \text{(Same as before except sum to } D') \\
&= 4\eta \frac{\sum_{i=1}^{D'} \lambda_i v_{di}^2 + \sum_{i'=D'+1}^D \lambda_{i'} v_{di}^2}{\sum_{i=1}^{D'} v_{di}^2} && (\lambda_{i'} = 0) \\
&= 4\eta \frac{\alpha_d}{P_{dd}} \\
\lim_{t \rightarrow \infty} \text{rate}_{dt}^{\text{non-ghost}} &= 1 - \lim_{t \rightarrow \infty} \frac{\sum_{i=1}^{D'} (1-2\eta\lambda_i)^{2t+2} v_{di}^2}{\sum_{i=1}^{D'} (1-2\eta\lambda_i)^{2t} v_{di}^2} \\
&\text{(No matter what the } v_{di}^2 \text{ are, both sums geometrically are dominated by the last term.)} \\
&= 1 - \frac{(1-2\eta\lambda_{\min}^+)^{2t+2} v_{dD'}^2}{(1-2\eta\lambda_{\min}^+)^{2t} v_{dD'}^2} \\
&= 1 - (1-2\eta\lambda_{\min}^+)^2 \\
&= 4\eta\lambda_{\min}^+ && \text{(Remove } 4\eta^2 \lambda_i^2) \\
\lim_{t \rightarrow \infty} \text{rate}_{dt}^{\text{all}} &= 0 \\
&\text{(\lambda}_{\min}^+ \text{ is the smallest non-zero eigenvalue; here we include the zero eigenvalues, giving rate 0.)}
\end{aligned}$$

□

Theorem 2.1.8 (Extra Facts).

1. $[\mathcal{E}_{\parallel i} = 0] \rightarrow [\mathbf{e} \text{ perpendicular to } \mathbf{v}_i]$.
2. $\mathcal{E} = \sum_d \mathcal{E}_d = \sum_d e_d^2 = \sum_i \mathcal{E}_{\parallel i}$
3. $\mathbb{E}[\mathcal{E}_{\parallel i}] = \frac{1}{D} \mathcal{E}$
4. $\mathbb{E}[\hat{\mathcal{E}}_d] \approx \frac{D'}{D} \mathcal{E}_d$ (randomly rotated $\mathbf{v}_i$)
5. $\text{Avg}_i \lambda_i = \frac{1}{D} \text{Trace}(\mathbf{R}) = \text{Avg}_d \alpha_d$
6. No Effect of Ghost on Non-Ghost
7. If for each i , $\Delta \mathcal{E}_{\parallel i} = r \mathcal{E}_{\parallel i}$, then for each d , $\Delta \mathcal{E}_d = r \mathcal{E}_d$.²³

Proof:

1. **Perpendicular:** Note that having a number of the spectral errors $\mathcal{E}_{\parallel i}$ zero just means that our error vector $\mathbf{e}$ is perpendicular to the subspace spanned by their eigenvectors $\mathbf{v}_i$.
2. $\mathcal{E} = \sum_d \mathcal{E}_d = \sum_d e_d^2 = \sum_i \mathcal{E}_{\parallel i}$: Being an orthonormal basis, lengths of vectors are preserved.
3. $\mathbb{E}[\mathcal{E}_{\parallel i}] = \frac{1}{D} \mathcal{E}$: Randomness in f_1 will give randomly rotated eigenvectors $\mathbf{v}_i$. Randomness in the targets y_d^* gives a random rotation of the error vectors $\mathbf{e}$. Either gives you the result.

4. $\mathbb{E}[\hat{\mathcal{E}}_d] = \frac{D'}{D} \mathcal{E}_d$: The ghost error is expected (with random rotations) to be distributed even among the point errors $\mathcal{E}_d$, giving that the ghost part $\mathcal{E}_d - \hat{\mathcal{E}}_d \approx \frac{k}{D} \mathcal{E}_d$ of each remains undecayed, while the non-ghost part $\hat{\mathcal{E}}_d \approx \frac{D'}{D} \mathcal{E}_d$ decays.
5. $\text{Avg}_i \lambda_i = \frac{1}{D} \text{Trace}(\mathbf{R}) = \text{Avg}_d \alpha_d$: The decay rate of $\mathcal{E}_{\parallel i}$ is λ_i and the decay rate of $\mathcal{E}_d$ is α_d . These two error components are weighted averages of each other, with the sum of squares of the weights one. Hence we expect these rates to be similar on average. The trace identity confirms this exactly. This identity is proved using the characteristic polynomial.
6. **No Effect of Ghost on Non-Ghost**: Having k zeros increases the average of the others from $\frac{1}{D} \sum_i \lambda_i$ to $\frac{1}{D-k} \sum_i \lambda_i$, but since each point error $\mathcal{E}_d$ mixes all spectral components, the frozen ghost fraction offsets this speedup.
Also because $\mathbf{R}e^{\text{ghost}} = \mathbf{0}$, ghost errors produce zero coupling noise.
7. Our assumption gives $\frac{\partial \mathcal{E}_{\parallel i}}{\partial t} = r \mathcal{E}_{\parallel i}$. By definition $\mathcal{E}_{\parallel i} = e_{\parallel i}^2$. Taking the square root, $\frac{\partial e_{\parallel i}}{\partial t} = \frac{1}{2e_{\parallel i}} \frac{\partial e_{\parallel i}^2}{\partial t} = \frac{1}{2e_{\parallel i}} r \mathcal{E}_{\parallel i} = \frac{1}{2} r e_{\parallel i}$. Since $e_d = \sum_i e_{\parallel i} v_{di}$ (change of basis), linearity of differentiation gives $\frac{\partial e_d}{\partial t} = \sum_i \frac{\partial e_{\parallel i}}{\partial t} v_{di} = \sum_i \frac{1}{2} r e_{\parallel i} v_{di} = \frac{1}{2} r e_d$. Squaring: $\frac{\partial \mathcal{E}_d}{\partial t} = r \mathcal{E}_d$.

□

12 Simulations: Eigenvector Structure of $\mathbf{R}$ and Per-Point Rates

This section proves no results but does simulations. The highlights were listed in the list of “theorems” as follows.

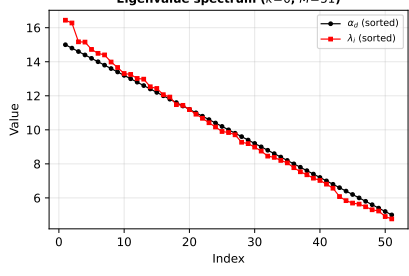
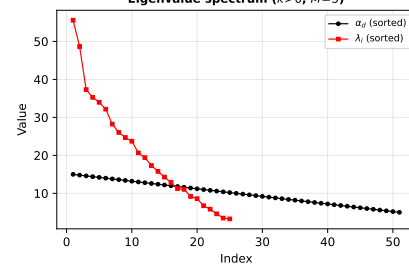
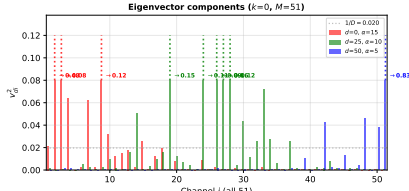
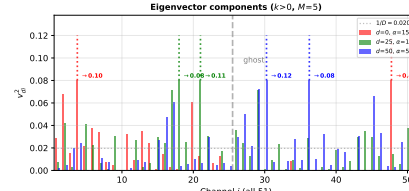
Theorem 2.2 (Simulations).

We run simulations on random $\mathbf{R}$ to understand what we could not prove about gradient descent. The insights not mirrored in the theoretical results are as follows:

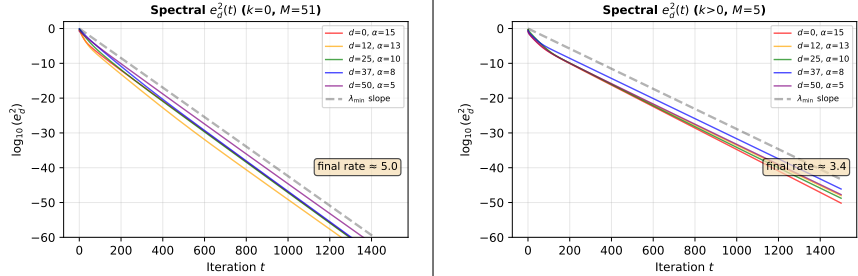
- **$\alpha_d \Leftrightarrow \lambda_{i_d}$** : When heavily overparameterized ($MM' \gg D$) each point d has a dedicated spectral channel i_d with $\lambda_{i_d} \approx \alpha_d$. This allows each point to decay at its own rate α_d .
- **Rotating $\mathbf{R}$** : We try randomly rotating $\mathbf{R}$ in order to move our current error into faster channels. As hoped, when $MM' < D$, this did manage to decay the ghost error. However, this did not keep the rate up near α_d or $\bar{\alpha}$ as hoped. The rate settled at $\lambda_{\min}^+$, which is even lower than $\alpha_{\min}$. (Section 12)
- **Gradient Descent**: When running actual gradient descent in this same $MM' < D$ range, we did not manage to get rid of the ghost error because either our rate matrices $\mathbf{R}$ refused to rotate, or $\lambda_{\min}^+$ dropped too close to zero.

For this discussion, independently for each point error $\mathbb{E}[\mathcal{E}_d] = 1$.

	Overparameterized ($D' = D$)	Rank-deficient ($D' < D$)
Rank of $\mathbf{R}$	Recall Theorem 2.6.3 gives that whp the $\text{Rank}(\mathbf{R}) = D' = \min(D, MM')$. $D = 51$, $M = M' = 51$, $MM' = 2601 \gg D$, $D' = 51$, $k = D - D' = 0$	$D = 51$, $M = M' = 5$, $MM' = 25 < D$, $D' = 25$, $k = D - D' = 26$

	Overparameterized ($D' = D$)	Rank-deficient ($D' < D$)
Sensitivity α_d Eigenvalue λ_i Fig 1.A and Fig 1.B		
	Plots sensitivity $\alpha(d)$ for each data point $d \in [D]$ and eigenvalue $\lambda(i)$ for each spectral channel $i \in [D]$.	
Sensitivity α_d	We set α_d to be linear from $\alpha_{\max} = 15$ to $\alpha_{\min} = 5$. We tried different curves $\alpha(d)$, but it did not make a big difference. We did want $\alpha_{\min} \gg 0$ so that $\lambda_{\min}^+ \gg 0$.	
Data Rep. $\mathbf{x}_d = f_1(\mathbf{a}_d)$ Steepest Dir. $\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d)$ Rate Matrix $\mathbf{R}$	In line with Theorem 2.6.4, we drew $\mathbf{X} \in \mathbb{R}^{M \times D}$ and $\mathbf{G} \in \mathbb{R}^{M' \times D}$ i.i.d. $\mathcal{N}(0, 1)$. We normalized $\ \mathbf{x}_d\ = 1$. We scale columns $\ \mathbf{g}_d\ ^2$ to be our chosen values α_d . We form the Khatri–Rao (Hadamard) matrix $r_{dd'} = (\mathbf{x}_d^T \mathbf{x}_{d'}) (\mathbf{g}_d^T \mathbf{g}_{d'})$.	
Eigendecomposition	We do a spectral decomposition $\mathbf{R} = \sum_i \lambda_i \mathbf{v}_i \mathbf{v}_i^T$.	
Ghost Channels	A spectral channel i is considered ghost if $\lambda_i = 0$. There will be $k = D - D'$ of these. Ghost Error e^{ghost} does not decay because $\mathbf{R} e^{\text{ghost}} = \lambda_i e^{\text{ghost}} = 0$. $k=0$. Every channel is alive. $k = D - D' = 51 - 25 = 26$ frozen. Fig 1.B is zero for last 26 channels.	
Eigenvalues λ_i	Fig 1 plots sorted $\lambda(i)$ in red. The trace identity $\sum_i \lambda_i = \sum_d \alpha_d$ includes the ghost errors, so the non-ghost average goes up by $\frac{D}{D'}$ when these are excluded. Each point d has a dedicated spectral channel i_d with $\lambda_{i_d} \approx \alpha_d$. Index i has no meaning so we sort by λ_i. By the pigeon hole principle, the D points cannot be mapped uniquely to the remaining D' non-ghost channels.³⁰ The Marchenko-Pastur distribution conjectured in “Theorem” 2.6.8 is confirmed within $\sim 1.5\%$ by simulation for two layers $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$. $\text{CV}(\lambda) := \frac{\text{SDV}(\lambda)}{\lambda_{\text{avg}}} = \sqrt{\frac{D}{MM'}}$	
Eigenvectors $\mathbf{v}_i$ Fig 2.A and Fig 2.B		

³⁰We saw some limited evidence in our simulations that the scaled $\lambda(i)$ curve tends to be between being a straight line and trying to mirror the shape of the $\alpha(d)$ curve.

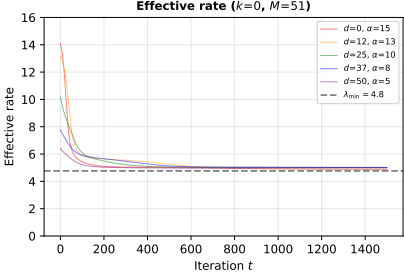
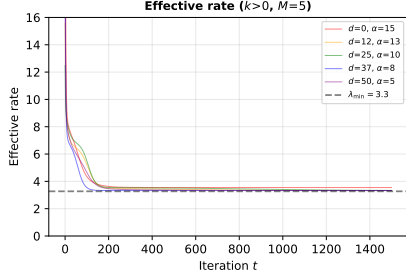
	Overparameterized ($D' = D$)	Rank-deficient ($D' < D$)
	Plots three points' eigen-element v_{di}^2 vs channel i . Recall $\sum_i v_{di}^2 = 1$. ³¹ Here point $d=0$ with $\alpha_0=15$ is in red, $d=25$ with $\alpha_{25}=10$ in green, and $d=50$ with $\alpha_{50}=5$ in blue.	
Fraction of Error	Point d 's error can be decomposed as $e_d = \sum_i e_{\parallel i} v_{di}$. Since $\mathbb{E}[e_{\parallel i} e_{\parallel i'}] = 0$ for $i \neq j$ (eigenvector orthogonality), the cross terms vanish in expectation, giving $\mathbb{E}[\mathcal{E}_d] = \sum_i \mathbb{E}[\mathcal{E}_{\parallel i}] v_{di}^2$. Hence v_{di}^2 gives the fraction of point d 's expected error in the i^{th} channel.	
Localized Eigenvectors:	Each point d clusters around a few channels close to their rate α_d ³² with weight $\sum_{i \approx i_d} v_{di}^2 \approx 0.50$ to 0.70 .	There are a few peaks here as well but in seemingly random places. Eg., red $\alpha_0 = 15$ has a high $v_{0,47}^2 = .46$ within the ghost errors. This means that this 46% of the error will never decay.
Delocalized:	The rest of the error weight v_{di}^2 is spread evenly between the channels. Randomly rotated eigenvectors have $ v_{di} \approx 1/\sqrt{D}$ (Theorem 2.6.7). ³³	
Error Decay Fig 3.A and Fig 3.B	 Plots three points' $\log_{10}(\mathbb{E}[\hat{\mathcal{E}}_{dt}]) = \log_{10}(\sum_{i=1}^{D'} (1-2\eta\lambda_i)^{2t} v_{di}^2)$ vs iteration t.	
Spectral Error	If the error vector $\mathbf{e} = [f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*]$ looks like $\mathbf{e} = \mathbf{v}_i$, then $\partial \mathbf{e} = \mathbf{R} \mathbf{e} = \lambda_i \mathbf{e}$. Hence, spectral error i 's decay is $\mathbb{E}[\mathcal{E}_{\parallel i}] = (1-2\eta\lambda_i)^{2t}$ $\mathbb{E}[\mathcal{E}_{\parallel i0}] = (1-2\eta\lambda_i)^{2t}$. $\log_{10}(\mathbb{E}[\hat{\mathcal{E}}_{dt}]) \approx -4\eta\lambda_i t$ giving the slope $\propto -\lambda_{\min} = -5$ in Fig 2. (Theorem 2.1.2)	
Non-Ghost:	Because ghost error does not decay we only graph the non-ghost. This means including only channels $i \in [1, \dots, D']$. Each point has a random fraction $P_{dd} = \sum_{i=1}^{D'} v_{di}^2$ of its error non-ghost. $\mathbb{E}[P_{dd}] = \frac{D'}{D}$ being the fraction of non-ghost channels. Points with large α_d tend to have larger P_{dd} . ³⁴	

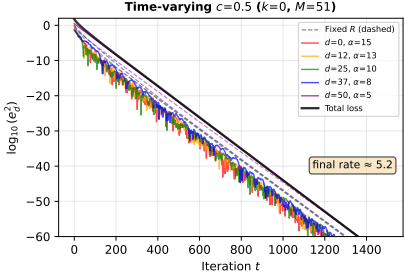
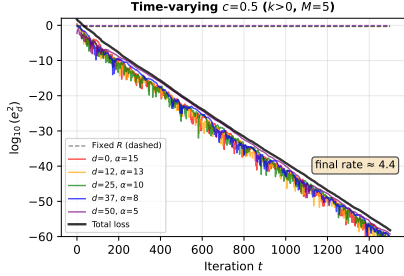
³¹Eigenvectors are orthonormal.

³²Meaning v_{di}^2 is big for i for which $\lambda_i \approx \alpha_d$.

³³We can't have $v_{di}^2 = 1/D$ exactly, otherwise $\alpha_d = r_{dd} = \sum_{i=1}^{D'} \lambda_i v_{di}^2 = \frac{1}{D} \bar{\lambda}$ would be constant.

³⁴From $\alpha_d = \sum_{i=1}^{D'} \lambda_i v_{di}^2 \leq \lambda_{\max} \cdot P_{dd}$, so $P_{dd} \geq \alpha_d / \lambda_{\max}$. High α_d forces more weight into non-ghost channels. Simulated: $\text{Corr}(\alpha_d, P_{dd}) \approx 0.66$.

	Overparameterized ($D' = D$)	Rank-deficient ($D' < D$)
Logarithmic	We take the logarithm because the t is in the exponent and because it gives a better impression of the total time needed. For example, suppose the fast channels make up 90% of the error. Removing it decreases the error by a factor of $\frac{1}{10}$ but only decreases $\log_{10}(\text{error})$ by an additive 1. In contrast, decreasing the error from 1 to $\epsilon = 10^{-6}$, requires decreasing $\log_{10}(\text{error})$ by another 6.	
Error Decay Fig 4.A and Fig 4.B	  Plots three points' rate_{dt} vs iteration t. $\text{rate}_{dt} = 1 - \frac{\mathbb{E}[\hat{\mathcal{E}}_{d(t+1)}]}{\mathbb{E}[\hat{\mathcal{E}}_{dt}]} = 1 - \frac{\sum_{i=1}^{D'} (1-2\eta\lambda_i)^{2t+2} v_{di}^2}{\sum_{i=1}^{D'} (1-2\eta\lambda_i)^{2t} v_{di}^2}.$	
Fig 3 -Slope = Fig 4 value	$-\frac{\partial \log(\mathbb{E}[\hat{\mathcal{E}}_{dt}])}{\partial t} = -\frac{1}{\mathbb{E}[\hat{\mathcal{E}}_{dt}]} \frac{\mathbb{E}[\hat{\mathcal{E}}_{d(t+1)}] - \mathbb{E}[\hat{\mathcal{E}}_{dt}]}{\partial t} = 1 - \frac{\mathbb{E}[\hat{\mathcal{E}}_{d(t+1)}]}{\mathbb{E}[\hat{\mathcal{E}}_{dt}]}$	
One Dominant Term	Suppose this sum is dominated by term i_*. $\mathbb{E}[\hat{\mathcal{E}}_{dt}] \approx (1-2\eta\lambda_{i_*})^{2t} v_{di_*}^2 \approx e^{-4\eta\lambda_{i_*} t} v_{di_*}^2. \quad \log(\mathbb{E}[\hat{\mathcal{E}}_{dt}]) \approx -4\eta\lambda_{i_*} t + c.$ $\text{rate}_{dt} = 1 - \frac{(1-2\eta\lambda_{i_*})^{2t+2} v_{di_*}^2}{(1-2\eta\lambda_{i_*})^{2t} v_{di_*}^2} = 1 - (1-2\eta\lambda_{i_*})^2 \approx 4\eta\lambda_{i_*}. \quad (\text{or simply } \lambda_{i_*}).$	
Initial Rate $\text{rate}_{d0} = 4\eta\alpha_d$	The initial Rate when considering all error is $\text{rate}_{d0}^{\text{all}} = 4\eta\alpha_d$ and when considering just non-ghost error is $\text{rate}_{d0}^{\text{non-ghost}} = 4\eta \frac{\alpha_d}{P_{dd}}$ (Theorem 2.1.7)	
High-α_d advantage is brief	The sum $\sum_{i=1}^{D'} (1-2\eta\lambda_i)^{2t} v_{di}^2$ is dominated by the $(1-2\eta\lambda_{i_d})^{2t} v_{di_d}^2$ term giving a rate of λ_{i_d} because the $v_{di_d}^2$ is so big. A few iterations t decays the error in fast channels (λ_i large) and then slowest channels ($\lambda_{\min}$) dominate.	Dies out even faster.
Rate Goes to $\text{rate}_{dt} = 4\eta\lambda_{\min}^+$	When the remaining channels have equal v_{di}^2, then the terms in the sum $\sum_{i=1}^{D'} (1-2\eta\lambda_i)^{2t} v_{di}^2$ grow geometrically as the λ_i get smaller. Hence, this is dominated by the $(1-2\eta\lambda_{\min}^+)^{2t} v_{di_{\min}}^2$ term giving a rate of $\lambda_{\min}^+$.	
Minimum $\lambda_{\min}^+$	Due to 1-1 mapping $\lambda_{\min}^+ = \alpha_{\min} = 5$.	$\lambda_{\min}^+ \approx \left(\sqrt{\frac{D}{MM'}} - 1 \right)^2 \bar{\alpha}$ $= \left(\sqrt{\frac{51}{5^2}} - 1 \right)^2 10 = 1.83. \quad \text{We got 1.15.}$
Running time	The stopping condition is that error reaches ϵ. See the dotted line in Fig 3 $\log_{10}(\epsilon) = \log_{10}(10^{-6}) = -6$ # iterations is between $\Theta(\alpha_d^{-1} \log(\mathcal{E}/\epsilon))$ and $O(\lambda_{\min}^+{}^{-1} \log(\mathcal{E}/\epsilon))$.	

	Overparameterized ($D' = D$)	Rank-deficient ($D' < D$)																					
Changing $\mathbf{R}$ Fig 5.A and Fig 5.B	 Time-varying $c=0.5$ ($k=0, M=51$) Same as Figs 4, except $\mathbf{R}$ changes each iteration. In gradient descent both $\mathbf{z}_d$ and $f_3$³⁵ change, which changes $\mathbf{g}_d$, $\mathbf{R}$, $\mathbf{V}$, and the spectral split, which moves error that had been slow or even ghost into other random places. Each iteration, we replace $c = 0.50$ fraction of the gradient $\mathbf{g}_{dt}$ with a random vector.³⁶ These varying matrices are combined as $\mathbf{e}_t = \prod_{t'=0}^{t-1} (\mathbf{I} - 2\eta\mathbf{R}_{t'}) \mathbf{e}_0$	 Time-varying $c=0.5$ ($k>0, M=5$)																					
Ghost Error Decays	No ghost error anyway	Fig 3.B only showed the non-ghost error, while Fig 5.B shows all error. Varying $\mathbf{R}$ makes the ghost error go away. The dashed line in Fig 5.B shows the fixed $\mathbf{R}$ plateauing at $10^{-0.3}$ with ghost error stuck.																					
Rate Goes to $\text{rate}_{dt} = 4\eta\alpha_{\min}$	Varying $\mathbf{R}$ makes very little change. Like with fixed $\mathbf{R}$, the rate quickly goes to $\alpha_{\min} = \lambda_{\min}^+ = 5$. (Here it spends less time with rate α_d)	Larger c moves the rate from from $\lambda_{\min}^+ = 3.3$ to $\alpha_{\min} = 5$. <table border="1"> <thead> <tr> <th>c</th><th>rate</th><th>final error</th></tr> </thead> <tbody> <tr> <td>0 (fixed)</td><td>stuck</td><td>2.6×10^1</td></tr> <tr> <td>0.10</td><td>0.40</td><td>5×10^{-6}</td></tr> <tr> <td>0.25</td><td>1.86</td><td>1.4×10^{-31}</td></tr> <tr> <td>0.50</td><td>4.25</td><td>2.2×10^{-74}</td></tr> <tr> <td>0.75</td><td>5.29</td><td>6.0×10^{-92}</td></tr> <tr> <td>1.00</td><td>5.14</td><td>1.6×10^{-91}</td></tr> </tbody> </table>	c	rate	final error	0 (fixed)	stuck	2.6×10^1	0.10	0.40	5×10^{-6}	0.25	1.86	1.4×10^{-31}	0.50	4.25	2.2×10^{-74}	0.75	5.29	6.0×10^{-92}	1.00	5.14	1.6×10^{-91}
c	rate	final error																					
0 (fixed)	stuck	2.6×10^1																					
0.10	0.40	5×10^{-6}																					
0.25	1.86	1.4×10^{-31}																					
0.50	4.25	2.2×10^{-74}																					
0.75	5.29	6.0×10^{-92}																					
1.00	5.14	1.6×10^{-91}																					
Disappointing	The author was hoping varying $\mathbf{R}$ would bring the rate up to $\bar{\lambda} = \bar{\alpha} = 10$ ³⁷ or even $\lambda_{id} \approx 15$, because each new $\mathbf{R}$ gives new channels and new distribution of error in those channels.																						
Funny Drops	These quick drops towards zero and return occur when the values of each terms does not change that much but which term dominates changes: $(\text{term}_{1,t} - \text{term}_{2,t})^2 \approx (\text{term}_{1,t})^2$ $(\text{term}_{1,(t+1)} - \text{term}_{2,(t+1)})^2 \approx (0)^2$ $(\text{term}_{1,(t+2)} - \text{term}_{2,(t+2)})^2 \approx (-\text{term}_{2,(t+2)})^2$																						

³⁵We don't change f_3 in this paper.

³⁶ $\mathbf{g}'_{d(t+1)} = (1-c) \frac{\mathbf{g}_{dt}}{\|\mathbf{g}_{dt}\|} + c \frac{\Delta \mathbf{g}_{dt}}{\|\Delta \mathbf{g}_{dt}\|}$ and $\mathbf{g}_{d(t+1)} = \sqrt{\alpha_d} \frac{\mathbf{g}'_{d(t+1)}}{\|\mathbf{g}'_{d(t+1)}\|}$.

³⁷By the trace identity: $\sum_{i=1}^D \lambda_i = \text{Trace}(\mathbf{R}) = \sum_{d=1}^D \alpha_d$

	Overparameterized ($D'=D$)	Rank-deficient ($D'<D$)																																																
Total $\mathcal{E}$ Higher	The total error $\mathcal{E} = \sum_{d=0}^{50} e_d^2$ being a sum is on average $D = 51$ times bigger and hence its $\log_{10}$ is 1.7 higher.																																																	
Gradient Descent Fails to Help	With no ghost error, this case converges to zero whatever we do, though the rate may be slow.	We tried a number of different architectures and parameters (soft-plus f_3 from Theorem 2.6, training only $\mathbf{W}$) and in each found one of two cases: Lazy: $\mathbf{R}$ remains mostly fixed so everything looks like Rows 1–4 and the ghost errors do not decay. Non-Lazy: $\mathbf{R}$ changes substantially, but many $\alpha_d \rightarrow 0$ (vanishing gradients), dragging $\lambda_{\min}^+ \rightarrow 0$ with them. The non-ghost rate crashes and ghost errors remain frozen. • The collapse of α_d for some data points is the neural-network analogue of vanishing gradients: data points whose $\alpha_d \rightarrow 0$ become invisible to the gradient and stop learning. • Real networks mitigate this with Adam, batch normalization, learning-rate schedules, etc.—techniques that stabilize α_d while still allowing $\mathbf{R}$ to evolve. Our time-varying walk (Row 5) models the idealized outcome: $\mathbf{g}_d$ changes enough to drain ghosts while α_d stays controlled.																																																
		Sorted by increasing change in $\mathbf{R}$. Final values of $\alpha_{\min}$, $\lambda_{\min}^+$, & Error almost sorted decreasing.																																																
	<table> <tr> <th>Activation</th> <th>MM'</th> <th>k</th> <th>$\mathbf{R}$ change</th> <th>$\alpha_{\min}$</th> <th>$\lambda_{\min}^+$</th> <th>Error</th> </tr> <tr> <td>ReLU</td> <td>2601</td> <td>0</td> <td>32%</td> <td>3.9</td> <td>2.9</td> <td>100%</td> </tr> <tr> <td>softplus</td> <td>25</td> <td>26</td> <td>50%</td> <td>3</td> <td>0.10</td> <td>5%</td> </tr> <tr> <td>softplus</td> <td>49</td> <td>2</td> <td>82%</td> <td>0.3</td> <td>0.0000</td> <td>34%</td> </tr> <tr> <td>ReLU</td> <td>25</td> <td>26</td> <td>98%</td> <td>0.8</td> <td>0.001</td> <td>50%</td> </tr> <tr> <td>softplus</td> <td>64</td> <td>0</td> <td>134%</td> <td>0.2</td> <td>0.003</td> <td>3%</td> </tr> <tr> <td>softplus</td> <td>36</td> <td>15</td> <td>314%</td> <td>0.0</td> <td>0.009</td> <td>6%</td> </tr> </table>	Activation	MM'	k	$\mathbf{R}$ change	$\alpha_{\min}$	$\lambda_{\min}^+$	Error	ReLU	2601	0	32%	3.9	2.9	100%	softplus	25	26	50%	3	0.10	5%	softplus	49	2	82%	0.3	0.0000	34%	ReLU	25	26	98%	0.8	0.001	50%	softplus	64	0	134%	0.2	0.003	3%	softplus	36	15	314%	0.0	0.009	6%
Activation	MM'	k	$\mathbf{R}$ change	$\alpha_{\min}$	$\lambda_{\min}^+$	Error																																												
ReLU	2601	0	32%	3.9	2.9	100%																																												
softplus	25	26	50%	3	0.10	5%																																												
softplus	49	2	82%	0.3	0.0000	34%																																												
ReLU	25	26	98%	0.8	0.001	50%																																												
softplus	64	0	134%	0.2	0.003	3%																																												
softplus	36	15	314%	0.0	0.009	6%																																												
The time-varying walk does not claim to replicate gradient descent. It models the best-case scenario: $\mathbf{g}_d$ randomizes enough to rotate the null space while $\ \mathbf{g}_d\ ^2 = \alpha_d$ stays fixed. Real feature learning lies between lazy (Fig 3) and full randomization (Fig 5).																																																		

13 Negligible Noise

This section restates and proves Theorem 2.3.

Theorem 2.3 (Negligible Noise).

The point-wise decay rate is $r_d = \alpha_d$ with negligible noise when either

	Random f_3	Arbitrary f_3
Random f_1	$M \geq \Omega\left(\sqrt{\frac{\alpha_{\max}}{\alpha_{\min}}} \frac{e_{\max}}{e_{\min}} D\right)^{23}$	$M \geq \Omega\left(\frac{\alpha_{\max}}{\alpha_{\min}} \frac{e_{\max}^2}{e_{\min}^2} D^2\right)$
Arbitrary f_1	$M' \geq \Omega\left(\frac{\alpha_{\max}}{\alpha_{\min}} \frac{e_{\max}^2}{e_{\min}^2} D^2\right)$ (Conjectured)	No. The noise dominates.

Status. This is a statement about one iteration. The width needed grows with $e_{\max}/e_{\min}$, which can grow during training, and for random f_3 the near-orthogonality of the $\mathbf{g}_d$ is known only at initialization. So it does not show that the rate $r_d = \alpha_d$ persists.

Conditions:

- **Random f_1 and $\mathbf{X}$:** These results use randomness in f_1 to ensure whp that the data representation $\mathbf{x}_d = f_1(\mathbf{a}_d)$ has randomly rotated eigenvectors $\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}} \mathbf{H}$. For example, the input could be a single real number $\mathbf{a}_d$, with f_1 consisting of one layer of activations followed by random weights.
- **Arbitrary f_3 (worst case):** The worst case for f_3 is when all gradient directions coincide: $\hat{g}_d = \hat{g}_{d'}$ for all d, d' . This gives $\mathbf{g}_d^T \mathbf{g}_{d'} = \sqrt{\alpha_d \alpha_{d'}}$, maximizing the off-diagonal terms $|r_{dd'}|$. A linear f_3 with identical $\mathbf{g}_d$ is one such case, but the bound holds for any f_3 whose gradients are collinear.
- **Random f_3 :** When f_3 is chosen randomly, the gradients are approximately pairwise orthogonal, decreasing $\mathbf{g}_d^T \mathbf{g}_{d'}$ and hence the noise.
- **Arbitrary f_1 (worst case):** This is listed as a conjecture because the proof covers delocalized eigenvectors ($v_{di}^2 \approx 1/D$) and nearly localized ones ($v_{di}^2 \approx 0/1$), but not the intermediate regime.
- **Trivial Case:** When f_1 is such that $\mathbf{X}^T \mathbf{X} = \mathbf{I}$, then $\mathbf{R} = \text{diag}(\alpha_d)$ and rates decouple trivially: each point decays independently at rate α_d with zero noise.

Proof. The goal is to show that the off-diagonal noise $\xi_d := \sum_{d' \neq d} |r_{dd'} e_{d'}|$ is at most half the diagonal signal $\alpha_d |e_d|$.

Case 1: Random f_1 , random f_3 . Both $\mathbf{x}_d \in \mathbb{R}^M$ and $\hat{g}_d \in \mathbb{R}^{M'}$ are random, so the near-orthogonality theorem applies to both:

$$\begin{aligned}
 |\mathbf{x}_d^T \mathbf{x}_{d'}| &\leq O\left(\sqrt{\frac{\log D}{M}}\right) && \text{(Thm 6.6: random unit } \mathbf{x}_d \text{ in } \mathbb{R}^M) \\
 |\hat{g}_d^T \hat{g}_{d'}| &\leq O\left(\sqrt{\frac{\log D}{M'}}\right) && \text{(Same theorem for unit } \hat{g}_d \text{ in } \mathbb{R}^{M'}) \\
 |r_{dd'}| &= \sqrt{\alpha_d \alpha_{d'}} \cdot |\hat{g}_d^T \hat{g}_{d'}| \cdot |\mathbf{x}_d^T \mathbf{x}_{d'}| && \text{(Factor: } \mathbf{g}_d = \sqrt{\alpha_d} \hat{g}_d, \|\mathbf{x}_d\| = 1) \\
 &\leq \sqrt{\alpha_d \alpha_{d'}} \cdot O\left(\frac{\log D}{M}\right) && \text{(Multiply the two bounds; set } M' = M)
 \end{aligned}$$

Set the bound to the target $\frac{\sqrt{\alpha_{\min}}}{\sqrt{\alpha_{\max}}} \frac{e_{\min}}{e_{\max}} \frac{1}{2D}$. This gives $M \geq \Omega\left(\sqrt{\frac{\alpha_{\max}}{\alpha_{\min}}} \frac{e_{\max}}{e_{\min}} D\right)$.

Case 2: Random f_1 , arbitrary f_3 . Only $\mathbf{x}_d$ is random; the worst case is $\hat{g}_d = \hat{g}_{d'}$ so $\mathbf{g}_d^T \mathbf{g}_{d'} = \sqrt{\alpha_d \alpha_{d'}}$.

$$\begin{aligned} |r_{dd'}| &= \sqrt{\alpha_d \alpha_{d'}} \cdot |\mathbf{x}_d^T \mathbf{x}_{d'}| && \text{(All gradient directions equal: } |\hat{g}_d^T \hat{g}_{d'}| = 1) \\ &\leq \sqrt{\alpha_d \alpha_{d'}} \cdot O\left(\sqrt{\frac{\log D}{M}}\right) && \text{(Only one near-orthogonality bound)} \end{aligned}$$

Set $\sqrt{\frac{\log D}{M}} = \frac{\sqrt{\alpha_{\min}}}{\sqrt{\alpha_{\max}}} \frac{e_{\min}}{e_{\max}} \frac{1}{2D}$. This gives $M \geq \Omega\left(\frac{\alpha_{\max}}{\alpha_{\min}} \frac{e_{\max}^2}{e_{\min}^2} D^2\right)$.³⁸

Case 3: Arbitrary f_1 , random f_3 . Only $\hat{g}_d \in \mathbb{R}^{M'}$ is random; $\mathbf{x}_d$ is arbitrary.

$$\begin{aligned} |r_{dd'}| &= \sqrt{\alpha_d \alpha_{d'}} \cdot |\hat{g}_d^T \hat{g}_{d'}| \cdot |\mathbf{x}_d^T \mathbf{x}_{d'}| && \text{(General factored form)} \\ &\leq \sqrt{\alpha_d \alpha_{d'}} \cdot O\left(\sqrt{\frac{\log D}{M'}}\right) && \text{(Worst case } |\mathbf{x}_d^T \mathbf{x}_{d'}| = 1; \text{ only } \hat{g} \text{ orthogonality)} \end{aligned}$$

Set $\sqrt{\frac{\log D}{M'}} = \frac{\sqrt{\alpha_{\min}}}{\sqrt{\alpha_{\max}}} \frac{e_{\min}}{e_{\max}} \frac{1}{2D}$. This gives $M' \geq \Omega\left(\frac{\alpha_{\max}}{\alpha_{\min}} \frac{e_{\max}^2}{e_{\min}^2} D^2\right)$.³⁹

Case 4: $\mathbf{X}^T \mathbf{X} = \mathbf{I}$. Then $\mathbf{x}_d^T \mathbf{x}_{d'} = 0$ for $d \neq d'$, so $r_{dd'} = 0$ and $\mathbf{R} = \text{diag}(\alpha_d)$. Each error decouples: $\Delta e_d = 2\eta \alpha_d e_d$, giving rate α_d with zero noise. $\square$

Shared conclusion (Cases 1–3). In each case we have arranged $|r_{dd'}| \leq \sqrt{\alpha_d \alpha_{d'}} \cdot \tau$ where $\tau = \frac{\sqrt{\alpha_{\min}}}{\sqrt{\alpha_{\max}}} \frac{e_{\min}}{e_{\max}} \frac{1}{2D}$.

$$\begin{aligned} \text{Signal: } r_{dd} e_d &= \alpha_d e_d && \text{(Def}^n \text{ of } \mathbf{R}'\text{'s diagonal with } \|\mathbf{x}_d\| = 1) \\ \text{Noise: } |r_{dd'} e_{d'}| &\leq \sqrt{\alpha_d \alpha_{d'}} \cdot \tau \cdot |e_{d'}| && \text{(Plug in bound from the relevant case)} \\ &\leq \frac{1}{2D} \alpha_d |e_d| && \text{(Since } \sqrt{\alpha_{d'}/\alpha_{\max}} \leq 1, \\ &&& |e_{d'}/e_{\max} \leq 1, \sqrt{\alpha_{\min}/\alpha_d} e_{\min}/|e_d| \leq 1) \\ \xi_d &:= \sum_{d' \neq d} |r_{dd'} e_{d'}| \leq \frac{1}{2} \alpha_d |e_d| && \text{(Sum } D-1 \text{ terms, each } \leq \frac{1}{2D} \alpha_d |e_d|) \\ \Delta e_d &= 2\eta \sum_{d'} r_{dd'} e_{d'} && \text{(Theorem 2.1.1)} \\ &\geq 2\eta \alpha_d e_d - 2\eta \sum_{d' \neq d} |r_{dd'} e_{d'}| && \text{(Triangle inequality)} \\ &\geq \frac{1}{2} \cdot 2\eta \alpha_d e_d. && \text{(Noise } \leq \text{half of signal)} \quad \square \end{aligned}$$

Remark. The quadratic improvement from Case 2 to Case 1 has a clean explanation: with collinear $\mathbf{g}_d$, only the $\mathbf{x}$ -side factor in $r_{dd'} = (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'})$ concentrates. With random $\mathbf{g}_d$, *both* factors are $O(\sqrt{\log D/M})$, and their product is $O(\log D/M)$ —each dimension of width M contributes orthogonality *twice*. Conversely, Case 3 is weaker than Case 2 because $|\mathbf{x}_d^T \mathbf{x}_{d'}|$ can reach 1 without random f_1 ; the $\alpha_{\max}/\alpha_{\min}$ penalty compensates for the lost cancellation.

³⁸Squaring the bound: squaring eliminates the $\sqrt{\cdot}$ on the α ratio (unlike Case 1 where two $\sqrt{\cdot}$ factors partially cancel), and also gives $D \rightarrow D^2$ and $e_{\max}/e_{\min} \rightarrow e_{\max}^2/e_{\min}^2$.

³⁹Without random f_1 , the $\mathbf{x}$ -side provides no concentration ($|\mathbf{x}_d^T \mathbf{x}_{d'}|$ can reach 1), so only $\hat{g}$ -orthogonality contributes. The bound matches Case 2 (with $M \rightarrow M'$) because each has exactly one concentrating factor.

14 Managed Noise

This section restates and proves Theorem 2.4.

Status. The $\pm$ bound of Lemma 1 is derived heuristically: its random-walk step treats signs as independent of the errors, and it uses near-orthogonality of the $\mathbf{g}_d$, which is known only at initialization. We therefore treat it as a hypothesis, the *coupling hypothesis* $|\xi_d| \leq \sqrt{\rho \alpha_d \bar{\alpha} \bar{\mathcal{E}}}$, with ρ not yet derived from the random-network model. Given it (and constant α_d), the “Bounding Collective” calculation below is valid: $\bar{\alpha} \bar{\mathcal{E}}$ decays at rate at least $\alpha_{\text{coll}}(1 - \sqrt{\rho \bar{\alpha} / \alpha_{\min}})$. The “Tracking the Threshold” argument is withdrawn.

Theorem 2.4 (Managed Noise).

*Initially, all points decay at their own rate α_d .*²⁰

When the fast ones are too far ahead, the cross-point noise drags everyone to

the collective rate $\alpha_{\text{coll}} = \frac{\sum_d \alpha_d^2 \mathcal{E}_d}{\sum_d \alpha_d \mathcal{E}_d}$ (average of α_d weighted by $\alpha_d \mathcal{E}_d$)^{24 25}

1. f_1 is random; f_3 is arbitrary; and $M \geq \frac{4\bar{\alpha}}{\alpha_{\min}} D$
2. f_1 is random; f_3 is random; and $MM' \geq \Omega\left(\frac{\bar{\alpha}}{\alpha_{\min}} D\right)$ ²³

For more on these conditions see Section 13.

We proceed as follows.

- Lemma 1 about signal vs noise is stated.
- Theorem 2.4 is proved from these.
- Lemma 1 is proved.

Lemma 1 (Per-Point Rate ($\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}} \mathbf{H}$)).

Assumption: Theorem 2.6.7 holds.

Decaying: Gradient descent decays each data point d ’s error $\mathcal{E}_d$ as follows:

$$\begin{aligned}
 \mathcal{E}_d \text{ Decay Rate:} \quad r_d &= \alpha_d \left(1 \pm \sqrt{\frac{\rho \bar{\alpha} \bar{\mathcal{E}}}{\alpha_d \mathcal{E}_d}} \right) \text{ (decreasing)} \\
 \bar{\alpha} \bar{\mathcal{E}} \text{ Decay Rate:} \quad r_{\text{coll}} &= \alpha_{\text{coll}} \left(1 \pm \sqrt{\frac{\rho \bar{\alpha}}{\alpha_{\min}}} \right) \\
 \text{Change:} \quad \Delta \mathcal{E}_d &= 4\eta r_d \mathcal{E}_d \text{ (Decrease)} \\
 \text{Recurrence:} \quad \mathcal{E}_d(t) &= (1 - 4\eta r_d)^t \mathcal{E}_{d0} \approx e^{-4\eta r_d t} \mathcal{E}_{d0} \\
 \# \text{ Iterations:} \quad T_d &= \Theta\left(\frac{1}{\eta r_d} \log \frac{\mathcal{E}_{d0}}{\epsilon}\right) \\
 &T_{\text{coll}} = \Theta\left(\frac{1}{\eta r_{\text{coll}}} \log \frac{\bar{\alpha} \bar{\mathcal{E}}_0}{\epsilon}\right)
 \end{aligned}$$

Where:

$$\begin{aligned}
 \text{Point Error}^2: \quad \mathcal{E}_d &= (f_3(\mathbf{W} \mathbf{x}_d) - y_d^*)^2 \\
 \text{Point Sensitivity:} \quad \alpha_d &= \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2 \\
 \text{Collective Rate:} \quad \alpha_{\text{coll}} &= \frac{\sum_d \alpha_d^2 \mathcal{E}_d}{\sum_d \alpha_d \mathcal{E}_d} \text{ (average of } \alpha_d \text{ weighted by } \alpha_d \mathcal{E}_d \text{)}^{24} \\
 \text{Phase Threshold:} \quad \alpha_d \mathcal{E}_d &\text{ vs } \rho \bar{\alpha} \bar{\mathcal{E}} \\
 \text{Constant:} \quad \rho &= \frac{D}{MM'}
 \end{aligned}$$

Proof of Theorem 2.4 (Managed Noise).

Each data point d converges in the following different ways depending on how its current error $\mathcal{E}_d$ compares to its threshold $\mathcal{E}_d^{\text{thresh}} = \frac{\rho \bar{\alpha} \bar{\mathcal{E}}}{\alpha_d}$:

- **Rate for those Above Threshold:** Points d above the threshold $\mathcal{E}_d \geq 2\mathcal{E}_d^{\text{thresh}} = \frac{2\rho \bar{\alpha} \bar{\mathcal{E}}}{\alpha_d}$, decay steadily at a rate of $r_d = \alpha_d \left(1 - \sqrt{\frac{\rho \bar{\alpha} \bar{\mathcal{E}}}{\alpha_d \mathcal{E}_d}} \right) > \alpha_d \left(1 - \sqrt{\frac{1}{2}} \right)$. (Lemma 1). The points in

this phase tend to have the following properties:

- **Many/Half Above:** We ensure $\rho \leq \frac{1}{4}$.⁴⁰ Rearranging the threshold requirement gives $\mathcal{E}_d \alpha_d \geq \frac{1}{2} \bar{\alpha} \bar{\mathcal{E}} \geq 2\rho \bar{\alpha} \bar{\mathcal{E}}$. Heuristically many/half of the data points should meet the requirement that their personal value $\mathcal{E}_d \alpha_d$ is at least half of the average.
- **At first Faster, then Slower:** At the beginning of the gradient descent, all the errors $\mathcal{E}_d$ will be roughly the same. Hence, the faster points (bigger α_d) will likely have bigger $\alpha_d \mathcal{E}_d$ values and hence be above the threshold. Faster points, however, will decay their errors $\mathcal{E}_d$ faster and hence their value $\alpha_d \mathcal{E}_d$ will quickly drop below average. Though to their advantage, these points do have lower threshold $\mathcal{E}_d^{\text{thresh}} = \frac{\rho \bar{\alpha} \bar{\mathcal{E}}}{\alpha_d} \propto \frac{1}{\alpha_d}$.
- **Rate for those Below Threshold:** Points d below the threshold $\mathcal{E}_d \leq \mathcal{E}_d^{\text{thresh}} = \frac{\rho \bar{\alpha} \bar{\mathcal{E}}}{\alpha_d}$, will have their error increase at a rate of $r_d = \alpha_d \left(1 - \sqrt{\frac{\rho \bar{\alpha} \bar{\mathcal{E}}}{\alpha_d \mathcal{E}_d}}\right) < 0$. But don't worry, (if not frozen) they will do alright in the end. [Not justified: the rate only lies in the $\pm$ range, whose lower end is negative. A point below its threshold can keep decreasing.]
- **Collective Rate:** Because $MM' \geq \frac{4\bar{\alpha}}{\alpha_{\min}} D$ and $\rho = \frac{D}{MM'}$, the average $\bar{\alpha} \bar{\mathcal{E}}$ decays at a steady rate of $r_{\text{coll}} = \alpha_{\text{coll}} \left(1 - \sqrt{\frac{\rho \bar{\alpha}}{\alpha_{\min}}}\right) \geq \alpha_{\text{coll}} \left(1 - \sqrt{\frac{1}{4}}\right)$. At this rate, it will take time $T_{\text{coll}} = \Theta\left(\frac{1}{\eta r_{\text{coll}}} \log \frac{\bar{\alpha} \bar{\mathcal{E}}_0}{\epsilon}\right)$ until $\bar{\alpha} \bar{\mathcal{E}} = \frac{1}{D} \sum_d \alpha_d \mathcal{E}_d \leq \epsilon$. This implies that on “average” $\mathcal{E}_d \leq \epsilon / \alpha_d$.
- **Outliers:** Outliers with very small $\alpha_{d'}$ raise the threshold for everyone. Theorem 2.7.2 lets us deem them frozen, lowering $\bar{\alpha} \bar{\mathcal{E}}$ for the active points.

□

Proof. (Lemma 1 Per-Point Rate $\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}} \mathbf{H}$) **Rate of Decay:**

$$\begin{aligned}
\Delta e_d &= 2\eta \sum_{d'} r_{dd'} e_{d'} && \text{(Theorem 2.1.1)} \\
&\quad \text{where } r_{dd'} = (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'}) && \text{(Definition)} \\
\mathbf{g}_d^T \mathbf{g}_d &= \|\mathbf{g}_d\|^2 = \alpha_d && \text{(Random } \hat{g}: \text{ near-orthogonality of } \mathbf{r}) \\
|\mathbf{g}_d^T \mathbf{g}_{d'}| &\leq \sqrt{\alpha_d \alpha_{d'}} \cdot O\left(\sqrt{\frac{\log D}{M'}}\right) && \text{(Random } \hat{g}: \text{ near-orthogonality of } \mathbf{r}) \\
\mathbf{x}_d^T \mathbf{x}_{d'} &= [\hat{\mathbf{V}} \hat{\Lambda} \hat{\mathbf{V}}^T]_{dd'} = \sum_i \hat{\lambda}_i \hat{v}_{di} \hat{v}_{d'i} && \text{(Eigendecomposition of } \mathbf{X}^T \mathbf{X})^{41} \\
&= \hat{\lambda}_{\text{avg}} \underbrace{\sum_i \hat{v}_{di} \hat{v}_{d'i}}_{\delta_{dd'}} + \sum_i (\hat{\lambda}_i - \hat{\lambda}_{\text{avg}}) \hat{v}_{di} \hat{v}_{d'i} && (\hat{\lambda}_{\text{avg}} \text{ splitting; orth. of } \hat{\mathbf{V}}) \\
\|\mathbf{x}_d\|^2 &= \hat{\lambda}_{\text{avg}} + \underbrace{\sum_i (\hat{\lambda}_i - \hat{\lambda}_{\text{avg}}) \hat{v}_{di}^2}_{\approx 0 \text{ for } \hat{v}_{di}^2 \approx 1/D} \approx \hat{\lambda}_{\text{avg}} && (d' = d; \text{ Theorem 2.6.7}) \\
\Delta e_d|_{\text{signal}} &= 2\eta r_{dd} e_d && \text{(Diagonal)} \\
&= 2\eta \|\mathbf{g}_d\|^2 \|\mathbf{x}_d\|^2 e_d = 2\eta \alpha_d \hat{\lambda}_{\text{avg}} e_d && \text{(Plugging in.)} \\
|\Delta e_d|_{\text{noise}} &= |2\eta \sum_{d' \neq d} r_{dd'} e_{d'}| \\
&\leq |2\eta \sum_{d' \neq d} \sqrt{\alpha_d \alpha_{d'}} \cdot O\left(\sqrt{\frac{\log D}{M'}}\right) (\sum_i (\hat{\lambda}_i - \hat{\lambda}_{\text{avg}}) \hat{v}_{di} \hat{v}_{d'i}) e_{d'}| && \text{(Plugging in; random-}\hat{g} \text{ bound)} \\
&= 2\eta \sqrt{\alpha_d} \left| \sum_i \hat{v}_{di} (\hat{\lambda}_i - \hat{\lambda}_{\text{avg}}) (\sum_{d' \neq d} \hat{v}_{d'i} \sqrt{\alpha_{d'}} e_{d'}) \right| && \text{(Rearrange)} \\
&= 2\eta \sqrt{\alpha_d} \left| \sum_i (\pm \frac{1}{\sqrt{D}}) (\hat{\lambda}_i - \hat{\lambda}_{\text{avg}}) (\sum_{d' \neq d} (\pm \frac{1}{\sqrt{D}}) \sqrt{\alpha_{d'}} e_{d'}) \right| && \text{(Theorem 2.6.7 } \hat{v}_{di} = \pm \frac{1}{\sqrt{D}})
\end{aligned}$$

⁴⁰Because $MM' \geq \frac{4\bar{\alpha}}{\alpha_{\min}} D$, we have $\rho = \frac{D}{MM'} \leq \frac{\alpha_{\min}}{4\bar{\alpha}} \leq \frac{1}{4}$.

$$\approx_{\text{whp}} 2\eta\sqrt{\alpha_d} \underbrace{\sqrt{\frac{1}{D} \sum_i (\hat{\lambda}_i - \hat{\lambda}_{\text{avg}})^2}}_{\text{SDV}(\hat{\lambda})} \underbrace{\sqrt{\frac{1}{D} \sum_{d' \neq d} \alpha_{d'} e_{d'}^2}}_{\sqrt{\alpha \mathcal{E}}} \quad (\text{Rand walk: } |\sum \pm a_i| \approx \sqrt{\sum a_i^2})$$

The $O(\sqrt{\log D/M'})$ max-over- d' bound at line 2 is valid but suboptimal for the sum: since $\hat{g}_d^T \hat{g}_{d'}$ are independent $O(1/\sqrt{M'})$ random variables across d' , the random walk concentrates without the extra $\sqrt{\log D}$. The constant $q=O(1)$ below absorbs the probabilistic bounds.

Noise to Signal Ratio:

$$\begin{aligned} \frac{|\Delta e_d|_{\text{noise}}}{\Delta e_d|_{\text{signal}}} &\approx \frac{q2\eta\sqrt{\alpha_d} \text{SDV}(\hat{\lambda}) \sqrt{\alpha \mathcal{E}}}{2\eta \alpha_d \hat{\lambda}_{\text{avg}} e_d} = \underbrace{\frac{q \text{SDV}(\hat{\lambda})}{\hat{\lambda}_{\text{avg}}}}_{\sqrt{\rho}} \sqrt{\frac{\alpha \mathcal{E}}{\alpha_d \mathcal{E}_d}} = \sqrt{\frac{\rho \alpha \mathcal{E}}{\alpha_d \mathcal{E}_d}} \\ \rho &= \left(\frac{q \text{SDV}(\hat{\lambda})}{\hat{\lambda}_{\text{avg}}} \right)^2 = \frac{D}{MM'} \quad (\text{See Theorem 2.6.8}) \\ \Delta e_d &= \Delta e_d|_{\text{signal}} \pm |\Delta e_d|_{\text{noise}} \\ &= \Delta e_d|_{\text{signal}} \left(1 \pm \frac{|\Delta e_d|_{\text{noise}}}{\Delta e_d|_{\text{signal}}} \right) \\ &= 2\eta \alpha_d \hat{\lambda}_{\text{avg}} e_d \left(1 \pm \sqrt{\frac{\rho \alpha \mathcal{E}}{\alpha_d \mathcal{E}_d}} \right) \quad (\text{We assume } \|\mathbf{x}_d\|=1 \text{ and hence } \hat{\lambda}_{\text{avg}} \approx 1 \text{ by normalization.}) \end{aligned}$$

Squaring the Error:

$$\begin{aligned} \mathcal{E}_d &:= e_d^2 && (\text{Squared Error}) \\ \frac{\Delta \mathcal{E}_d}{\Delta t} &= 2e_d \frac{\Delta e_d}{\Delta t} && (\text{Calculus}) \\ \Delta \mathcal{E}_d &= 2e_d \Delta e_d = 4\eta \alpha_d \hat{\lambda}_{\text{avg}} \left(1 \pm \sqrt{\frac{\rho \alpha \mathcal{E}}{\alpha_d \mathcal{E}_d}} \right) \mathcal{E}_d \end{aligned}$$

While $\alpha_d \mathcal{E}_d > \rho \alpha \mathcal{E}$, the ratio is less than 1 and signal dominates.

Bounding Collective:

- $\Delta \alpha \mathcal{E} = 4\eta r_{\text{coll}} \overline{\alpha \mathcal{E}}$ (Decrease)
- $r_{\text{coll}} = \alpha_{\text{coll}} \left(1 \pm \sqrt{\frac{\rho \alpha}{\alpha_{\min}}} \right)$
- Where $\alpha_{\min}$ is the smallest non-zero gradient.
- For ReLU activations, $\alpha_d = |\mathbf{g}_d|^2$ counts the number of active output neurons, so $\alpha_d \in \{0, 1, \dots, M'\}$.
- This bound is nearly tight. For two equal-sized groups under Phase 1 dynamics $\mathcal{E}_d(t) = (1 - 4\eta \alpha_d)^t$, the peak ratio converges to $\sqrt{\rho \alpha_{\max} / \alpha_{\min} + 1}/2$ as $\alpha_{\max} \rightarrow \infty$.

$$\begin{aligned} \Delta \mathcal{E}_d &= 4\eta (\alpha_d \left(1 \pm \sqrt{\frac{\rho \alpha \mathcal{E}}{\alpha_d \mathcal{E}_d}} \right)) \mathcal{E}_d && (\text{Proved above}) \\ \Delta \alpha_d \mathcal{E}_d &= 4\eta (\alpha_d^2 \left(1 \pm \sqrt{\frac{\rho \alpha \mathcal{E}}{\alpha_d \mathcal{E}_d}} \right)) \mathcal{E}_d && (\text{We multiply by } \alpha_d \text{ and assume (as an approximation) that this does not change with gradient descent.}) \\ \Delta \overline{\alpha \mathcal{E}} &= \frac{1}{D} \sum_d 4\eta (\alpha_d^2 \left(1 \pm \sqrt{\frac{\rho \alpha \mathcal{E}}{\alpha_d \mathcal{E}_d}} \right)) \mathcal{E}_d && (\text{The derivative of the sum is the sum of the derivative}) \\ &= \text{signal} \pm \text{noise} && (\text{Define}) \\ &= \frac{\text{signal}}{\alpha \mathcal{E}} \left(1 \pm \frac{\text{noise}}{\text{signal}} \right) \overline{\alpha \mathcal{E}} && (\text{Rearranged}) \end{aligned}$$

⁴¹Throughout this section, $\hat{\lambda}_i$ and $\hat{v}_{di}$ denote the eigenvalues and eigenvectors of the Gram matrix $\mathbf{X}^T \mathbf{X}$, distinct from the (unhatted) λ_i, v_{di} used for $\mathbf{R}$'s eigendecomposition in the spectral section.

$$\begin{aligned}
&= 4\eta \alpha_{\text{coll}} \left(1 \pm \sqrt{\frac{\rho \bar{\alpha}}{\alpha_{\min}}}\right) \overline{\alpha \mathcal{E}} && \text{(Will be computed below)} \\
&= 4\eta r_{\text{coll}} \overline{\alpha \mathcal{E}} && \text{(Definition of } r_{\text{coll}} \text{ gives result)} \\
\frac{\text{signal}}{\overline{\alpha \mathcal{E}}} &= \frac{\frac{1}{D} 4\eta \sum_d \alpha_d^2 \mathcal{E}_d}{\frac{1}{D} \sum_d \alpha_d \mathcal{E}_d} = 4\eta \alpha_{\text{coll}} && \text{(Definition of signal, } \overline{\alpha \mathcal{E}}, \text{ and } \alpha_{\text{coll}}) \\
\frac{\text{noise}}{\text{signal}} &= \frac{\frac{1}{D} \sum_d 4\eta (\alpha_d^2 \left(\sqrt{\frac{\rho \bar{\alpha}}{\alpha_d \mathcal{E}_d}}\right) \mathcal{E}_d}{\frac{1}{D} \sum_d 4\eta (\alpha_d^2) \mathcal{E}_d} && \text{(Definition of signal and noise)} \\
&= \frac{\sqrt{\rho \bar{\alpha \mathcal{E}}} \left[\sum_d \alpha_d^{3/2} \mathcal{E}_d^{1/2} \right]}{\sum_d \alpha_d^2 \mathcal{E}_d} && \text{(Simplifying)} \\
&\leq \frac{\sqrt{\rho \bar{\alpha \mathcal{E}}} \left[(\sum_d \alpha_d^2 \mathcal{E}_d)^{1/2} (\sum_d \alpha_d)^{1/2} \right]}{\sum_d \alpha_d^2 \mathcal{E}_d} && \text{(Cauchy-Schwarz with } a_d = \alpha_d \mathcal{E}_d^{1/2} \text{ and } b_d = \alpha_d^{1/2}) \\
&= \sqrt{\frac{\rho \bar{\alpha} \sum_d \alpha_d \mathcal{E}_d}{\sum_d \alpha_d^2 \mathcal{E}_d}} && \text{(Simplify)} \\
&\leq \sqrt{\rho \frac{\bar{\alpha} \sum_d \alpha_d \mathcal{E}_d}{\alpha_{\min} \sum_d \alpha_d \mathcal{E}_d}} && \text{(Factor out smallest non-zero value } \alpha_{\min}) \\
&= \sqrt{\frac{\rho \bar{\alpha}}{\alpha_{\min}}} && \text{(Cancel Sum)} \quad \square
\end{aligned}$$

Corollary 3 (Arbitrary/Linear f_3).

When f_3 is arbitrary (including linear f_3 where all $\hat{g}_d$ coincide), the random-direction bound $|\hat{g}_d^T \hat{g}_{d'}| \leq O(\sqrt{\log D/M'})$ is replaced by the Cauchy-Schwarz bound $|\mathbf{g}_d^T \mathbf{g}_{d'}| \leq \sqrt{\alpha_d \alpha_{d'}}$. The width requirement becomes $M \geq \frac{4\bar{\alpha}}{\alpha_{\min}} D$ with $\rho = \frac{D}{M}$.

Proof. Follow the proof of Lemma 1, replacing the random-direction bound $|\mathbf{g}_d^T \mathbf{g}_{d'}| \leq \sqrt{\alpha_d \alpha_{d'}} \cdot O(\sqrt{\log D/M'})$ with the Cauchy-Schwarz bound $|\mathbf{g}_d^T \mathbf{g}_{d'}| \leq \sqrt{\alpha_d \alpha_{d'}}$. The factor $\sqrt{\log D/M'}$ drops from the noise, giving $\rho = \frac{D}{M}$, and the rest of the proof is identical. $\square$

15 Random Walks (An Instructive Failed Approach)

This section restates and proves Theorem 2.5. A random-walk argument initially looked promising (signal $\alpha_d e_d$ is steady; noise ξ_d has random sign). This implies a random walk where the signal grows with T and the noise with $\sqrt{T}$. We prove that even with adversarial dependence between iterations the decay occurs as we want with reasonable probability, provided the noise ξ_d has no bias toward the sign of the error e_d . The method fails because $\text{sign}(\xi_d)$ correlates with $\text{sign}(e_d)$. This effect arises from the ghost mechanism.

Theorem 2.5.1.a (Random Noise Sign).

At each iteration t , the noise has zero mean ($\mathbb{E}[\xi_{dt}] = 0$ and symmetric), giving it a random $\pm$ sign.

Proof. The noise $\xi_d = 2\eta \sum_{d' \neq d} (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'}) e_{d'} \in \mathbb{R}$ has three potential sources of randomness (any one suffices for $\mathbb{E}[\xi_d] = 0$).

- **Random $\mathbf{x}_d^T \mathbf{x}_{d'}$:** Each entry x_{md} within the data representation $\mathbf{X} \in \mathbb{R}^{M \times D}$ is independently normally distributed with mean zero (Theorem 2.6.4) over randomness in the network's fixed feature map f_1 . If $M < D$, then there is dependence between the points d . However, when computing the sum $\xi_d = 2\eta \sum_{d' \neq d} (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'}) e_{d'}$ we can use the fact that the expectation of the sum is the sum of the expectation.

- **Random $\mathbf{g}_d^T \mathbf{g}_{d'}$:** If the f_3 inputs $\mathbf{z}_{dt} \in \mathbb{R}^{M'}$ are sufficiently distinct, then the gradients $\mathbf{g}_{dt} = \nabla_{\mathbf{z}} f_3(\mathbf{z}_{dt}) \in \mathbb{R}^{M'}$ have independent normally distributed entries $\mathbf{g}_{m'dt}$. (Theorem 2.6.4)
- **Random $e_{d'}$:** We argue that the non-ghost component of the point-wise error $e_{dt} = f_3(\mathbf{z}_{dt}) - y_d^*$ is independently normally distributed, so its $\pm$ sign is random.
 - **Initial Errors:** Suppose everything about gradient descent (input and network weights) were fixed adversarially, except for the targets y_d^* . We could choose these targets so that the point errors $e_{dt} = f_3(\mathbf{z}_{dt}) - y_d^*$ are independently normally distributed with mean zero.
 - **Ongoing Errors:** Consider gradient descent as the following game. f_3 is chosen randomly. The goal of the GD player is to learn a value $\mathbf{z}_d$ such that $f_3(\mathbf{z}_d) = y_d^*$ for given targets y_d^* . Each iteration t , the player is allowed to give a vector $\mathbf{z}_{dt}$ and in response learns a barely useful fact $\nabla_{\mathbf{w}} \mathcal{E} = \sum_d (\nabla_{\mathbf{z}_d} \mathcal{E}_d) \mathbf{x}_d^T$. The player considers the distribution on f_3 conditioned on this information over the previous iterations. As in any 20-questions game, the optimal strategy would be to choose the next query $\mathbf{z}_d$ so that the sign of $f_3(\mathbf{z}_{dt}) - y_d^*$ were independently 50/50 random.
 - **Adversarial Dependence:** Theorem 2.5.3 shows that this argument fails because the error concentrates on ghost errors, which do not decay.

□

Theorem 2.5.1.b and 2 (Probability of Success).

Random $\pm$ sign on the noise implies a random walk where the signal grows with T and the noise with $\sqrt{T}$.

The sign of the noise ξ_{dt} , though random at each iteration, likely has huge dependence across iterations. For example, if nothing changes then the noise does not change $\xi_{dt} = \xi_{d(t-1)}$. This does mean, however, with probability $\frac{1}{2}$ all the noise is helpful. Hence, you only need to run the gradient descent twice to get the results you want. We generalize this idea to handle more complex adversarial dependencies.

Setup: Let's simplify the notation.⁴² Let r denote the additive per-iteration signal decay rate, ξ magnitude of the noise, and $\mu_t \in \{-1, 1\}$ its sign. The progress after T iterations is given by $P(T) = rT + (\sum_t \mu_t) \xi$. Assume $\forall t \Pr[\mu_t = 1] = \frac{1}{2}$. What is harder to argue is that these events are independent. The next lemma proves that even without this, the probability of success is at least $\frac{r}{4\xi}$.

Formal Statement:

- **Classic Random Walk:** If each sign μ_t is independently random, then

$$\mathbb{E}[|\sum_t \mu_t|] = \Theta(\sqrt{T}) \text{ and } \mathbb{E}[P(T)] \geq rT - \xi\sqrt{T}.$$

$$\text{When } T \geq \left(\frac{2\xi}{r}\right)^2, \mathbb{E}[P(T)] \geq \frac{1}{2}rT.$$

- **Worst Case Dependence:** If $\forall t \Pr[\mu_t = 1] = \frac{1}{2}$, then

$$\Pr[\sum_t \mu_t \geq -2\epsilon T] \geq \epsilon$$

$$\Pr[P(T) = rT + (\sum_t \mu_t) \xi \geq \frac{1}{2}rT] = \Pr[\sum_t \mu_t \geq -\frac{r}{2\xi}T] \geq \frac{r}{4\xi}$$

⁴²Because random walks are generally additive not multiplicative, we do the following transformation. Having the change in error be proportional to the current error $\partial e_d = 2\eta \alpha_d e_d$ leads to geometric decay. Namely, $e_{dt} = e_{d(t-1)} - \partial e_d = (1 - 2\eta \alpha_d)^t e_{d0} \approx e^{-2\eta \alpha_d t} e_{d0}$. Taking the logarithm gives additive decay $\ln(e_{dt}) = \ln(e_{d0}) - 2\eta \alpha_d t$. We then define $r = 2\eta \alpha_d$ and $\xi = |\partial \ln(e_d) - r|$. Note we drop the index d and assume that these do not change over iterations t .

This is tight.⁴³

Proof.

Let A denote the event that $\sum_t \mu_t < -2\epsilon T$.

By the contrapositive, assume that $\Pr_{\mathcal{D}}[\bar{A}] < \epsilon$.

Let $\mathcal{D}'$ denote the distribution $\mathcal{D}$ conditioned on A .

By Claim 1, there exists $t \in [T]$, such that $\Pr_{\mathcal{D}'}[\mu_t = 1] \leq \frac{1}{2} - \epsilon$.

Then $\Pr_{\mathcal{D}}[\mu_t = 1] = \Pr_{\mathcal{D}}[\mu_t = 1|A]\Pr_{\mathcal{D}}[A] + \Pr_{\mathcal{D}}[\mu_t = 1|\bar{A}]\Pr_{\mathcal{D}}[\bar{A}] < (\frac{1}{2} - \epsilon) \cdot 1 + 1 \cdot \epsilon = \frac{1}{2}$.

Claim 1: Consider any possibly dependent distribution $\mathcal{D}'$ on $\mu_t \in \{-1, 1\}$.

$[\Pr_{\mathcal{D}'}[\sum_t \mu_t < -2\epsilon T] = 1] \Rightarrow [\exists t \Pr_{\mathcal{D}'}[\mu_t = 1] \leq \frac{1}{2} - \epsilon]$.

Proof of Claim 1: Assume $\Pr_{\mathcal{D}'}[\sum_t \mu_t < -2\epsilon T] = 1$.

$$\begin{aligned} \mathbb{E}[\mu_t] &= 2\Pr[\mu_t = 1] - 1 && \text{(Definition)} \\ \sum_t (2\Pr[\mu_t = 1] - 1) &= \sum_t \mathbb{E}[\mu_t] = \mathbb{E}[\sum_t \mu_t] \leq -2\epsilon T && \text{(Linearity; a.s. } < c \Rightarrow \mathbb{E} \leq c) \\ \sum_t \Pr[\mu_t = 1] &\leq T(\frac{1}{2} - \epsilon) && \text{(Rearranging)} \\ \exists t \Pr[\mu_t = 1] &\leq \frac{1}{2} - \epsilon && \text{(Averaging / pigeonhole)} \end{aligned}$$

□

Theorem 2.5.3 (Adversarial Dependence).

The statement and proof appear at the beginning of Part II.

16 Random Network Structures

This section restates and proves all of the sub-results in Theorem 2.6. These results capture the properties arising from f_1 and f_3 being sufficiently complex and random. For example, f_1 could map an input consisting of a single real number $\mathbf{a}_d$ to one layer of activations followed by a fully connected layer $\mathbf{W}' \in \mathbb{R}^{M \times M}$ of random weights.

Theorem 2.6.1 (Rank of $\mathbf{X}$). *For random f_1 with sigmoids, $\mathbf{X} \in \mathbb{R}^{M \times D}$ has full rank $\min(D, M)$.*

With ReLU, columns may coincide (see Theorem 2.6.5).

Proof.

An Example f_1 Model/Architecture: We denote the early part of the neural network as f_1 . Its job is to take the d^{th} data point input $\mathbf{a}_d$ and produce its representation $\mathbf{x}_d = f_1(\mathbf{a}_d) \in \mathbb{R}^M$. Let us make this network more concrete.

Sigmoid Activations:

$$a_d \in \mathbb{R}$$

$$\omega \& \omega' \in \mathbb{R}^M$$

$$\mathbf{x}'_{md} = \text{Sig}(\omega_m a_d - \omega'_m) \in \mathbb{R}$$

$$\mathbf{W}' \in \mathbb{R}^{M \times M}$$

Worst case input type is a single real number.

$\omega_m \in \mathbb{R}$ is a random mult. and add. weights
from single input to the $m^{th} \in [M]$ node.

Node's post (sigmoid) activation values.

A fully connected layer of random weights

⁴³**Worst Case Distribution:** With probability ϵ , set all $\mu_t = 1$. Otherwise, independently for each $t \in [T]$, let $\mu_t = 1$ with probability $q = (\frac{1}{2} - \epsilon)(1 - \epsilon)^{-1} \approx \frac{1}{2} - \frac{1}{2}\epsilon$. Note that $\Pr[\mu_t = 1] = \epsilon + (1 - \epsilon)q = \epsilon + (\frac{1}{2} - \epsilon) = \frac{1}{2}$. Conditioned on the independent case, $\mathbb{E}[\sum_t \mu_t] = qT - (1 - q)T \approx -\epsilon T$ and $\Pr_{\mathcal{D}_\epsilon}[\sum_t \mu_t < -2\epsilon T] \leq \epsilon$. Hence, without the condition, $\Pr_{\mathcal{D}_\epsilon}[\sum_t \mu_t < -2\epsilon T] \leq 2\epsilon$.

$$\mathbf{x}_d = \text{Sig}(\mathbf{W}' \mathbf{x}'_d) \in \mathbb{R}^M$$

$$\mathbf{X} = [\mathbf{x}_d \mid d \in [D]] \in \mathbb{R}^{M \times D}$$

ReLU Activations:

$$\mathbf{x}'_{md} = \text{ReLU}(\omega_m a_d - \omega'_m) \in \mathbb{R}$$

$$\mathbf{x}_d = \text{ReLU}(\mathbf{W}' \mathbf{x}'_d) \in \mathbb{R}^M$$

$\mathbf{x}_d = f_1(\mathbf{a}_d)$ is the representation of the data considered in this paper.

If the weights ω_m and inputs a_d are distinct, the first data representation, which is $\mathbf{X}' = [\text{Sig}(\omega_m a_d - \omega'_m)]_{md} \in \mathbb{R}^{M \times D}$, has full rank $\min(D, M)$. Since the sigmoid is analytic and non-polynomial, this rank is a standard property of Chebyshev systems; see also Pinkus [1999] and the Vandermonde determinant argument for polynomial features. Multiplying it by a random matrix $\mathbf{W}' \in \mathbb{R}^{M \times M}$ and taking another sigmoid keeps this rank. $\square$

Theorem 2.6.2' ($\forall \mathbf{Z}$ Random f_3 Rank of $\mathbf{G}$).

For fixed pairwise distinct $\mathbf{z}_1, \dots, \mathbf{z}_D \in \mathbb{R}^{M'}$ and random f_3 , $\mathbf{G} \in \mathbb{R}^{M' \times D}$ has full rank $\min(D, M')$ with probability one over the random weights of f_3 .

Proof.

An Example f_3 Model/Architecture: The last part f_3 of the neural network produces the scalar output $f_3(\mathbf{z}_d)$. Let us make this network more concrete.

$\mathbf{z}_d \in \mathbb{R}^{M'}$ $\hat{\mathbf{z}}_d = \text{Sig}(\mathbf{z}_d) \in \mathbb{R}^{M'}$ $\omega, \omega' \in \mathbb{R}^{M'}$ $\mathbf{W}' \in \mathbb{R}^{M' \times M'}$ $f_3(\mathbf{z}_d) = \omega^T \text{Sig}(\mathbf{W}' \hat{\mathbf{z}}_d + \omega') \in \mathbb{R}$	Input to f_3 . Component-wise activation. Output weights and bias A fully connected layer of random weights Output.
---	---

Distinct $\mathbf{z}_d$ give distinct $\hat{\mathbf{z}}_d = \text{Sig}(\mathbf{z}_d)$ (the sigmoid is strictly monotone). Define $\mathbf{S} \in \mathbb{R}^{M' \times D}$ by $S_{jd} = \text{Sig}'(\mathbf{w}'_j^T \hat{\mathbf{z}}_d + \omega'_j)$, where $\mathbf{w}'_j$ is the j -th row of $\mathbf{W}'$. This matrix plays the role of $\mathbf{X}'$ in the Rank of $\mathbf{X}$ proof: it evaluates an analytic, non-polynomial function (Sig' in place of Sig) at random linear functionals of distinct inputs ($\hat{\mathbf{z}}_d$ in place of a_d), with biases ω'_j . By the same Chebyshev/Pinkus argument, $\mathbf{S}$ has full rank $\min(D, M')$ with probability one over the random $(\mathbf{W}', \omega')$.

By the chain rule, $g_{m'd} = \text{Sig}'(z_{m'd}) \cdot [\mathbf{W}'^T \text{diag}(\omega) \mathbf{S}]_{m'd}$. The square matrix $\mathbf{W}'^T \text{diag}(\omega)$ is invertible with probability one (both factors have nonzero determinant for random weights), so $\text{Rank}(\mathbf{W}'^T \text{diag}(\omega) \mathbf{S}) = \text{Rank}(\mathbf{S}) = \min(D, M')$. Each factor $\text{Sig}'(z_{m'd}) > 0$ is a fixed positive constant (depending on the given $\mathbf{z}_d$, not on the random weights), so the entry-wise scaling cannot make all maximal minors vanish.^{44 45} $\square$

⁴⁴Each minor determinant $\det(\mathbf{G}_I) = \sum_{\sigma} \text{sgn}(\sigma) \prod_i \text{Sig}'(z_{m'_i, d_{\sigma(i)}}) \hat{G}_{m'_i, d_{\sigma(i)}}$ is a multilinear form in the entries of $\hat{\mathbf{G}} = \mathbf{W}'^T \text{diag}(\omega) \mathbf{S}$ with all nonzero coefficients ($\prod_i \text{Sig}'(\cdot) > 0$ for every permutation σ). A polynomial with all nonzero coefficients and distinct monomials is not identically zero.

⁴⁵Without biases, Sig' may be symmetric ($\text{Sig}'(z) = \text{Sig}'(-z)$, for example), so setting $\mathbf{z}_{d'} = -\mathbf{z}_d$ would give $\text{Sig}'(\mathbf{w}^T \hat{\mathbf{z}}_d) = \text{Sig}'(\mathbf{w}^T \hat{\mathbf{z}}_{d'})$ for every $\mathbf{w}$, making the columns of $\mathbf{G}$ identical. With biases, the symmetry breaks: $\text{Sig}'(\mathbf{w}^T \hat{\mathbf{z}}_d + b) = \text{Sig}'(\mathbf{w}^T \hat{\mathbf{z}}_{d'} + b)$ requires either $\mathbf{w}^T(\hat{\mathbf{z}}_d - \hat{\mathbf{z}}_{d'}) = 0$ (a measure-zero hyperplane since $\hat{\mathbf{z}}_d \neq \hat{\mathbf{z}}_{d'}$) or, for symmetric Sig' , $b = -\mathbf{w}^T(\hat{\mathbf{z}}_d + \hat{\mathbf{z}}_{d'})/2$ (one bad bias value).

Changing $\mathbf{Z}$: After the random f_3 has been fixed, gradient descent proceeds, it changes $\mathbf{W}$, $\mathbf{z}_d = \mathbf{W}\mathbf{x}_d$, $\mathbf{g}_d := \nabla_{\mathbf{z}} f_3(\mathbf{z}_d)$, and $\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$ to unpredictable values. We need to understand the conditions under which $\mathbf{G}$ and $\mathbf{R}$ remain full rank.

Theorem 2.6.2 (Random $f_3 \forall \mathbf{Z}$ Rank of $\mathbf{G}$).

For fixed random f_3 , the set of pairwise-distinct $(\mathbf{z}_1, \dots, \mathbf{z}_D)$ for which $\mathbf{G}$ has rank less than $\min(D, M')$ is a measure-zero analytic subvariety of $\mathbb{R}^{M' \times D}$.

Proof. For each size- $\min(D, M')$ minor of $\mathbf{G}$, the determinant $h(\mathbf{Z}) = \det(\mathbf{G}_I(\mathbf{Z}))$ is an analytic function of $\mathbf{Z}$ (since Sig' is analytic). By Theorem 2.6.2', there exist pairwise distinct $\mathbf{z}_1, \dots, \mathbf{z}_D$ at which $\mathbf{G}$ has full rank, so h is not identically zero. The zero set of a non-identically-zero analytic function on $\mathbb{R}^{M' \times D}$ has Lebesgue measure zero. $\square$

$\mathbf{R} = (\mathbf{G}^T \mathbf{G}) \odot (\mathbf{X}^T \mathbf{X})$ has Khatri–Rao structure (Hadamard product of Gram matrices).

Theorem 2.6.3 (Rank of $\mathbf{R}$). For fixed random f_3 , the set of pairwise-distinct $(\mathbf{z}_1, \dots, \mathbf{z}_D)$ for which $\mathbf{R} \in \mathbb{R}^{D \times D}$ has rank less than $\min(D, MM')$ is a measure-zero analytic subvariety of $\mathbb{R}^{M' \times D}$.

(The Jacobian $\mathbf{J} \in \mathbb{R}^{D \times MM'}$ with $\mathbf{J}_{d, MM'} = \frac{\partial f_{\mathbf{W}}(\mathbf{a}_d)}{\partial \mathbf{W}_{MM'}}$ also has rank D since $\mathbf{R} = \mathbf{J}\mathbf{J}^T$.)

Proof. The rate matrix has entries $r_{dd'} = (\mathbf{g}_d^T \mathbf{g}_{d'}) \cdot (\mathbf{x}_d^T \mathbf{x}_{d'})$, which is the Hadamard product of two Gram matrices. By the Hadamard rank bound, $\text{Rank}(\mathbf{R}) \leq \text{Rank}(\mathbf{G}^T \mathbf{G}) \cdot \text{Rank}(\mathbf{X}^T \mathbf{X}) \leq M' \cdot M$. Since $\mathbf{R}$ is $D \times D$, $\text{Rank}(\mathbf{R}) \leq \min(D, MM')$; for tightness, each maximal minor of $\mathbf{J} = [\mathbf{g}_d \otimes \mathbf{x}_d]_d$ is an analytic function of the random weights in f_1 and f_3 . By Theorems 2.6.1 and 2.6.2, there exist weights at which $\mathbf{J}$ has full rank, so these analytic functions are not identically zero and their common zero set has measure zero. $\square$

Theorem 2.6.4 (Approximately Normal $\mathbf{X}$ and $\mathbf{G}$).

For random f_1 , each entry is $x_{md} \approx \mathcal{N}(0, \frac{1}{M})$ (columns may be correlated).

For random f_3 , each entry is $g_{m'd} \approx \mathcal{N}(0, \frac{1}{M'})$ (columns may be correlated).

Proof.

- **Sum of Independent Terms:** Suppose the data representation $\mathbf{X}'$ is linearly independent and each entry $w'_{mm'}$ of $\mathbf{W}'$ is chosen randomly according to its own independent distribution $\mathcal{D}_{mm'}$. By the Central Limit Theorem (Lindeberg version), each pre-activation entry $[\mathbf{W}'\mathbf{x}'_d]_m = \sum_{m'} w'_{mm'} x'_{m'd}$ is approximately normally distributed for large M , because it is the sum of M independent terms. If the weights have common mean $\mu = 0$ and variance $\sigma^2 = \frac{1}{M}$, then $\text{Mean}([\mathbf{W}'\mathbf{x}'_d]_m) = \mu \sum_{m'} x'_{m'd} = 0$ and $\text{Var}([\mathbf{W}'\mathbf{x}'_d]_m) = \sigma^2 \|\mathbf{x}'_d\|^2 = \frac{1}{M}$ giving the required $\mathcal{N}(0, 1/M)$.
- **Rows Independent:** Each row of $\mathbf{X}$ is *fully* independent of the others because the rows of $\mathbf{W}'$ are mutually independent. Knowing all the other rows tells you nothing about the m^{th} .
- **Columns Dependent:** When the inputs $\mathbf{a}_d$ and $\mathbf{a}_{d'}$ are too close, for a fixed neuron m , the entries $x_{md} = \text{Sig}(\mathbf{w}'_m \mathbf{x}'_d)$ and $x_{md'} = \text{Sig}(\mathbf{w}'_m \mathbf{x}'_{d'})$ share the weight row vector $\mathbf{w}'_m$ and are therefore correlated with covariance $\sigma^2 \mathbf{x}'_d{}^T \mathbf{x}'_{d'}$.
- **Next Sigmoid:** After sigmoid activation, the entries are bounded and no longer Gaussian, but satisfy similar sub-Gaussian concentration.

- **Adversarial game:** A difference with **G** is that we are considering f_3 changing across iterations t . The approximate way to do this is simply to apply in parallel the proof of $g_{m'dt} \approx \mathcal{N}(0, \frac{1}{M})$ for all of the inputs $\mathbf{z}_{dt}$ that f_3 ever receives over points d and iterations t . To be fair, however, an adversary might try to beat us by playing the following interactive game. Her goal is to make the entries $g_{m'dt}$ not approximately normally distributed. Initially, f_3 is chosen randomly but is unknown to the adversary. Before each iteration t , she knows only the barely useful gradient updates $\nabla_{\mathbf{w}} \mathcal{E}_{t'} = \sum_d g_{m'dt'} e_{dt'} \mathbf{x}_{dt'}^T$ learned during previous iterations $t' < t$. She considers the distribution on f_3 conditioned on this information. She is then allowed to give a vector $\mathbf{z}_{dt}$ for each d , with a requirement that they are all distinct. She wins if $g_{m'dt}$ fails to be approximately normal. Though in the multi-round game the adversary is able to adapt using this extra information, we conjecture that one could prove theoretically that this is not useful. Certainly in practice, it is unlikely an issue. □

Theorem 2.6.5 (Coupon Collector). *For random f_1 with sigmoid or ReLU activation functions and $M \geq \Omega(D \log D)$. Coupon Collector ensures each consecutive a_d vs a_{d+1} is split. Rank of $\mathbf{X}$ is D . Columns are approximately independent.*

Proof.

- **ReLU:** If $a_d = c a_{d'}$ and no ReLU splits between them, then $\mathbf{x}_d = c \mathbf{x}_{d'}$.
In contrast, suppose the input values $\mathbf{a}_d$ are sufficiently far apart and the distribution on weights ω_m is aimed to hit between these input values. Then whp coupon collector ensures each consecutive $\mathbf{a}_d$ vs $\mathbf{a}_{d+1}$ is split by the ReLU of some node $m \in [M]$. Meaning $\mathbf{x}'_{md} = \text{ReLU}(a_d - \omega_m) = \max(0, a_d - \omega_m) = 0$ and $\mathbf{x}'_{m,d+1} = \text{ReLU}(a_{d+1} - \omega_m) \neq 0$. The resulting matrix has triangular structure and is therefore full rank.
- **Sigmoids:** The same can be done here but the output values would be $\approx 0/1$.
- **Columns Sequentially Independent:** We have seen already that full rank of $\mathbf{X}'$ alone is not enough to get column independence. The triangular structure of $\mathbf{X}'$ is essential. The proof structure is as follows. Suppose the upper-triangular matrix $\mathbf{X}'$ is known and the random weight matrix $\mathbf{W}'$ is not. Define $\mathbf{X}'' := \mathbf{W}' \mathbf{X}'$ to be the pre-activation matrix so that $\mathbf{X} = \text{Sig}(\mathbf{X}'')$. An entry x''_{md} is said to be independent if it still has a good distribution conditioned on knowing the other entries. So suppose we know all of $\mathbf{X}''$ except the entry x''_{md} . Let us concentrate on row m , solving $x''_{md'} = \sum_j w'_{mj} x'_{jd'}$ for columns left to right as done in Gaussian elimination. The column $d' = d-1$ before ours and ours d are as follows.

Col/Row	Solving $x''_{md'} = \sum_j w'_{mj} x'_{jd'}$ column d' by column:
	$j < d'$ $j = d'$ $j = d$ $j > d$
$d' = d-1$	$ \underbrace{x''_{md'}}_{\text{known}} = \sum_{\substack{j < d' \\ \text{determined by} \\ \text{earlier columns}}} \underbrace{w'_{mj}}_{\text{determined}} \underbrace{x'_{jd'}}_{\text{newly } \neq 0} + \underbrace{w'_{md'}}_{\text{not used}} \underbrace{x'_{d'd'}}_{\neq 0} + \underbrace{w'_{md}}_{\text{not used}} \underbrace{x'_{dd'}}_0 + \sum_{j > d} \underbrace{w'_{mj}}_{\text{0 by upper triangular } \mathbf{X}'} \underbrace{x'_{jd'}}_{\text{0 by upper triangular } \mathbf{X}'} $

$$d \quad \left| \begin{array}{l} x''_{md} = \sum_{\substack{j < d' \\ \text{determined by} \\ \text{earlier columns}}} \underbrace{w'_{mj}}_{\text{determined by earlier columns}} x'_{jd} + \underbrace{w'_{md'}}_{\text{previously } \neq 0} \underbrace{x'_{d'd}}_{\text{used for } \neq 0} + \underbrace{w'_{md}}_{\text{used for } \neq 0} x'_{dd} + \sum_{j > d} \underbrace{w'_{mj}}_{\text{0 by upper triangular } \mathbf{X}'} x'_{jd} \\ \text{unknown} \end{array} \right.$$

The value of w'_{md} is an independent random variable that has not been fixed by the rest of $\mathbf{X}''$ and hence provides one degree of freedom for x''_{md} to be independent. The index md of w'_{md} and x''_{md} being the same helps with our bookkeeping that the same weight is not used more than once.

- **Next ReLU:** Zeros out the negative half, producing a *rectified normal* distribution. $\square$

Theorem 2.6.6 (Near-Orthonormal). For random f_1 , $\mathbf{X}^T \mathbf{X} \approx \mathbf{I}$ with $\mathbf{x}_d^T \mathbf{x}_{d'} = O\left(\sqrt{\frac{\log D}{M}}\right)$ for $d \neq d'$.

Proof. The way to get the data point $\mathbf{x}_d \in \mathbb{R}^M$ to be a unit vector pointing in a random direction is to independently choose each entry x_{md} according to the Normal distribution with mean zero and variance $\frac{1}{M}$ so that $\mathbb{E}[\|\mathbf{x}_d\|^2] = \sum_{m=1}^M \mathbb{E}[x_{md}^2] = M \left(\frac{1}{M}\right) = 1$. The square of a normally distributed variable has the chi-squared distribution χ^2 , which gives that the variance of this result is $\text{Var}(\|\mathbf{x}_d\|^2) = \sum_{m=1}^M \text{Var}(x_{md}^2) = M \cdot \frac{2}{M^2} = \frac{2}{M}$ and the standard deviation is $\sqrt{\frac{2}{M}}$. By χ^2 concentration: $P(|\|\mathbf{x}_d\|^2 - 1| > \xi) \leq 2 \exp\left(-\frac{M\xi^2}{8}\right)$.

For $d \neq d'$, the terms $x_{md}x_{md'}$ are i.i.d. with mean 0 and variance $\text{Var}(x_{md}x_{md'}) = \mathbb{E}[x_{md}^2] \mathbb{E}[x_{md'}^2] = \frac{1}{M^2}$ (by independence). Hence $\mathbb{E}[\mathbf{x}_d^T \mathbf{x}_{d'}] = 0$ and $\text{Var}(\mathbf{x}_d^T \mathbf{x}_{d'}) = \frac{M}{M^2} = \frac{1}{M}$. Since each summand is sub-exponential, the sum is sub-Gaussian and satisfies $P(|\mathbf{x}_d^T \mathbf{x}_{d'}| > \xi) \leq 2 \exp(-cM\xi^2)$ for a universal constant $c > 0$.

Setting $\xi = O\left(\sqrt{\frac{\log D}{M}}\right)$, the union bound gives that the probability that none of these at most D^2 bad events happens is at least $1 - O(D^{-1})$. $\square$

Theorem 2.6.7 (Randomly Rotated Eigenvectors). $\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}} \mathbf{H}$ (any M vs D):

For random f_1 , eigenvectors of $\mathbf{X}^T \mathbf{X}$ satisfy $\hat{v}_{di} \approx \text{Norm}(0, \frac{1}{D})$.

We would love to plug in $\pm \frac{1}{D}$ for $\hat{v}_{di}$, but likely some of them will be as big as $\pm \frac{\log D}{D}$, which we do not want plugged in for all of them. Instead:

Fix $s \leq D^{\Theta(1)}$ vectors $\mathbf{q} \in \mathbb{R}^D$ depending on $\hat{\mathbf{V}}$ only through entries not being summed.⁴⁶

Whp the following 4sDT sums are true simultaneously.

- $\forall d \quad \sum_i \hat{v}_{di}^2 q_i = (1 \pm \epsilon) \frac{1}{D} \sum_i q_i$
- $\forall d \quad |\sum_i \hat{v}_{di} q_i| \leq (1 + \epsilon) \sqrt{\frac{1}{D} \sum_i q_i^2}$
- Again with the role of d and i switched
- Again for each iteration $t \in [T]$.

⁴⁶Some test vectors $\mathbf{q}$ may depend on the eigenvalues $\hat{\lambda}_i$. This is compatible with independence from $\hat{\mathbf{V}}$ because for rotationally invariant ensembles (GOE, Wishart), eigenvalues and eigenvector directions are independent Mehta [2004], Anderson et al. [2010]. Some test vectors $\mathbf{q} = (\hat{\lambda}_i - \hat{\lambda}_{\text{avg}}) \mathbf{c}_i$ depend on $\hat{\mathbf{V}}$ through $c_i = \sum_{d'} \hat{v}_{d'i} \sqrt{\alpha_{d'}} e_{d'}$. This is OK because c_i depends on column entries $\hat{v}_{d'i}$ while the outer sum $\sum_i \hat{v}_{di} q_i$ uses row entries $\hat{v}_{di}$, and for Haar-random $\hat{\mathbf{V}}$ these are approximately independent.

Remark: In contrast, the eigenvectors of $\mathbf{R}$ are localized.⁴⁷

Proof.

- **Normal:** [Knowles and Yin, 2013, Bloemendal et al., 2014, O’Rourke et al., 2016] Suppose $\mathbf{X}$ has independence across rows and each entry is centered, with variance $1/M$, and bounded fourth moment. Then for any eigenvector $\hat{\mathbf{v}}_i$ of $\mathbf{X}^T\mathbf{X}$ (with nonzero eigenvalue), $\hat{v}_{di}$ is roughly $\mathcal{N}(0, 1/D)$.
- **Ghost:** When $M < D$, only $\hat{D} = M$ eigenvalues are nonzero. The corresponding $\hat{D}$ eigenvectors remain delocalized ($\hat{v}_{di}^2 \approx 1/D$ for $i \leq \hat{D}$), but the $D - \hat{D}$ ghost eigenvectors ($\hat{\lambda}_i = 0$) are arbitrary vectors that span the null space. As such we can choose them so that they too are delocalized.
- **Boolean:** Suppose the magnitude $|\hat{v}_{di}| = \frac{1}{\sqrt{D}}$ is fixed and each $\pm$ sign is chosen independently. Then the first sum is trivially true by factoring out the constant and the second is a standard random-walk concentration bound.
- $\hat{\mathbf{V}} \approx \frac{1}{\sqrt{D}}\mathbf{H}$: We use this notation because $\mathbf{H}$ is a Hadamard matrix – a square matrix filled with ± 1 where all rows are mutually orthogonal.
- **Normally Distributed:** Randomness in the feature map f_1 produces randomness in the data representation $\mathbf{X}$, which in turn produces randomness in the eigenvectors $\hat{\mathbf{V}}$. Theorem 2.6.7 then gives that each $\hat{v}_{di}$ is independently normally distributed, $\text{Norm}(0, \frac{1}{D})$. Some of these D^2 values $|\hat{v}_{di}|$ will be $\Theta(\sqrt{\log D})$ times bigger than $\frac{1}{\sqrt{D}}$. But this is ok, because we only care about the sum of D items.
- **Union Bound:** For any one of these $4sDT$ sums, the probability is $e^{-\Omega(\epsilon^2 D)}$ that it is off by more than this ϵ amount. The union bound gives us that probability that one of this fails is $4sDT \cdot e^{-\Omega(\epsilon^2 D)} \leq e^{-\Omega(\epsilon^2 D)}$.

□

Theorem 2.6.8 (Eigenvalues - Marchenko–Pastur).

For random f_1 , eigenvalues of $\mathbf{X}^T\mathbf{X}$ satisfy $\text{CV}(\hat{\lambda}) := \frac{\text{SDV}(\hat{\lambda})}{\hat{\lambda}_{\text{avg}}} = \sqrt{\frac{D}{M}}$.

Proof. Bai and Silverstein [2010] Thm. 3.6. Let $\mathbf{X}$ be an $M \times D$ matrix whose entries are i.i.d. with mean zero and variance σ^2/M . Let $\hat{\lambda}_1, \dots, \hat{\lambda}_D$ be the eigenvalues of the $D \times D$ Gram matrix $\mathbf{X}^T\mathbf{X}$. Fix $\gamma = \frac{D}{M} > 0$ and take the limit as D and M increase. $\hat{\lambda}_{\text{avg}} := \text{Avg}_i \hat{\lambda}_i = \sigma^2$. $\text{CV}(\hat{\lambda}) := \frac{\text{SDV}(\hat{\lambda})}{\hat{\lambda}_{\text{avg}}} = \sqrt{\gamma} = \sqrt{\frac{D}{M}}$. The mean follows from $\text{Avg}_i \hat{\lambda}_i = \frac{1}{D} \text{tr}(\mathbf{X}^T\mathbf{X}) = \frac{1}{D} \sum_{m,d} x_{md}^2 \rightarrow \sigma^2$ by the law of large numbers. The variance is a standard moment of the Marchenko–Pastur density. □

Theorem 2.6.8' ($\mathbf{R}$ Eigenvalues). For random f_3 :

Proven:

- $\bar{\lambda} = \bar{\alpha}$ and $\bar{\lambda}^+ = \frac{D}{D'} \bar{\alpha}$ (fewer channels share the trace budget $D\bar{\alpha}$).
- $D' = \min(D, MM')$ and $k = D - D'$ zero eigenvalues (ghost channels) follows from item 3.

Conjectures: Verified to 1.5% across all tested values.

⁴⁷Each eigenvector loads primarily on one data point.

Proof (orthogonal-data case): When $\mathbf{X}^T\mathbf{X} = \mathbf{I}$ (achievable for $M \geq D$), $R_{dd'} = (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'}) = \delta_{dd'} \|\mathbf{g}_d\|^2 = \delta_{dd'} \alpha_d$. So $\mathbf{R} = \text{diag}(\alpha_d)$ with eigenvalues exactly α_d and eigenvectors $\mathbf{e}_d$ —maximally localized. This is why per-point rates α_d survive as meaningful quantities rather than collapsing to $\bar{\alpha}$.

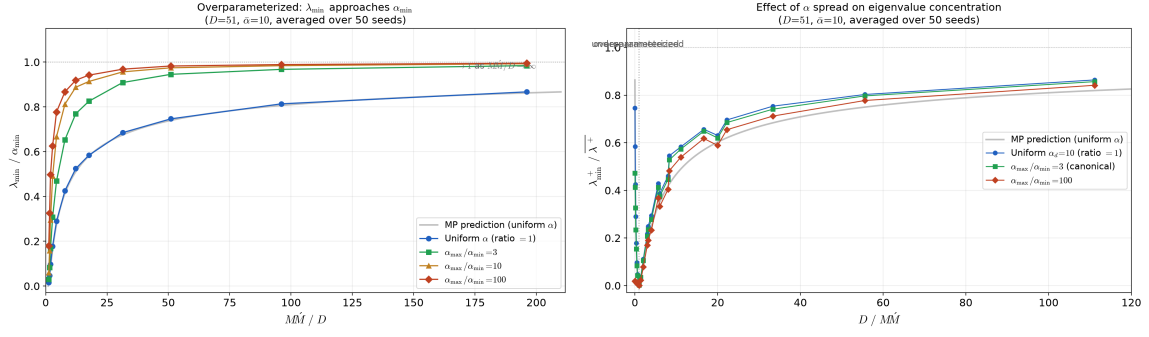


Figure 2: Supporting the Conjecture stated in Theorem 2.6.8'

- $CV(\lambda) \approx \sqrt{D/MM'}$: Simulations confirm this for the CV of all D eigenvalues (including zeros). The CV of only the D' non-zero eigenvalues is different.
- $MM' \gg D$: There is a 1-1 mapping between data points and spectral channels with $\lambda_{i_d} \approx \alpha_d$. In particular, $\lambda_{\min} \approx \alpha_{\min}$. See Figure 2.A.
- $MM' \approx D$: $\lambda_{\min}^+ \approx (\sqrt{D/MM'} - 1)^2 \bar{\alpha}$
This dips dangerously close to zero, stalling the decay of the “non-ghost” errors.
This can be “fixed”, however, by just calling such near-zero channels ghost.
- $MM' \ll D$: No matter how diverse the data point sensitivities α_d are, the non-zero eigenvalues of $\mathbf{R}$ are tightly concentrated around $\bar{\lambda}^+ = \frac{D}{D'} \bar{\alpha}$. Hence, the minimum non-zero eigenvalue is $\lambda_{\min}^+ \approx c \frac{D}{MM'} \bar{\alpha}$ for $c \geq \frac{1}{2}$ and 1 in the limit. See Figure 2.B.

“Proof”:

- $\bar{\lambda} = \bar{\alpha}$: Trace identity: $\bar{\lambda} = \frac{1}{D} \text{Tr}(\mathbf{R}) = \frac{1}{D} \sum_d \alpha_d = \bar{\alpha}$.
- The average of the D' non-zero eigenvalues is $\bar{\lambda}^+ = \frac{1}{D'} \sum_{i=1}^{D'} \lambda_i = \frac{D}{D'} \frac{1}{D} \sum_{i=1}^D \lambda_i = \frac{D}{D'} \bar{\lambda} = \frac{D}{D'} \bar{\alpha}$.
- $CV(\lambda) \approx \sqrt{D/(MM')}$: Empirical.
- The Marchenko–Pastur standard deviation law gives
 $\lambda_{\min}^+ \approx (1 - \sqrt{D/(MM')})^2 \alpha = (\sqrt{D/(MM')} - 1)^2 \alpha$.
 We wrote α here because this result implicitly assumes all the α_d are the same. Hence, we need to adjust this to be either $\bar{\alpha}$ or $\alpha_{\min}$.
- As MM' grows much bigger than D , the off-diagonals of $\mathbf{R}$ vanish, $\mathbf{R} \rightarrow \text{diag}(\alpha_d)$, $\lambda_{i_d} \rightarrow \alpha_d$, and in particular, $\lambda_{\min} \rightarrow \alpha_{\min}$. Our simulation of this is graphed in Figure 2.A and for $M \gg D$ proved in Theorem 2.3. The bound $(1 - \sqrt{D/(MM')})^2 \alpha$ goes to α . We make this correct by plugging in $\alpha_{\min}$ for α . We don’t write the “+” in $\lambda_{\min}^+$ because here there are no zero eigenvalues.
- When $MM' \leq D$, Figure 2(B) graphs the MP ratio $\frac{(\sqrt{D/MM'} - 1)^2}{D/MM'}$ wrt $\frac{D}{MM'}$. It also graphs the values of $\frac{\lambda_{\min}^+}{\bar{\lambda}^+}$ from our simulations. This nicely shows that these two graphed values are quite close. Assuming this, our result follows:

$$\begin{aligned}
 \lambda_{\min}^+ &\approx \frac{(\sqrt{D/MM'} - 1)^2}{D/MM'} \bar{\lambda}^+ && \text{(The two graphed values are close; rearranged.)} \\
 &= \frac{(\sqrt{D/MM'} - 1)^2}{D/MM'} \left(\frac{D}{MM'} \bar{\alpha} \right) && \text{(Plugging in the proven } \bar{\lambda}^+ = \frac{D}{MM'} \bar{\alpha} \text{.)} \\
 &= (\sqrt{D/MM'} - 1)^2 \bar{\alpha}.
 \end{aligned}$$

We repeated this simulating for a range of $\alpha_{\max}/\alpha_{\min}$ from 1 to 100 and $\lambda_{\max}^+/\lambda_{\min}^+$ changed by less than a factor of 2.

- When $MM' \ll D$, we have $(\sqrt{D/MM'} - 1)^2 \bar{\alpha} \approx \frac{D}{MM'} \bar{\alpha}$. This latter, however, is provably $\bar{\lambda}^+$. On the positive side, this indicates that the $\frac{D}{MM'}$ factor is correct. On the negative side, unless all the non-zero eigenvalues are exactly the same, it cannot also be $\lambda_{\min}^+$. Our simulations give that for $\frac{D}{MM'} \geq 11$, $c = \frac{\lambda_{\min}^+}{\lambda^+} \geq \frac{1}{2}$, concentrating the eigenvalues together. $\square$

17 Extensions and Generalizations

The preceding theorems make simplifying assumptions. This section shows when simplifying assumptions can be relaxed: Equal $\hat{\lambda}_i$; Ignoring Stalled Points; Worst case Weights $\mathbf{W}$; Second-Order Effects; Generalizing to $\langle \alpha, \beta \rangle$ -PL Sensitivity; Removing Redundant Rows.

17.1 Between the Lazy (NTK) and Feature-Learning

The regime for Part II is in the middle between the Lazy Training and Feature-Learning Regimes. It does not learn features as the M features of data point d in $\mathbf{x}_d := f_1(\mathbf{a}_d) \in \mathbb{R}^M$ are produced by a fixed pre-processing function f_1 . The post-processing function is also fixed. It is not Lazy because the rate matrix $\mathbf{R}_t = (\mathbf{G}_t^T \mathbf{G}_t) \odot (\mathbf{X}^T \mathbf{X})$ changes each iteration. It is not NTK because f_3 is highly nonlinear, not approximately linear.

	Lazy (NTK)	Part II Middle Ground	Feature Learning
Width	$M \geq \text{poly}(D)$ (strongly overpar.)	$MM' \geq \Omega(D)$ (mildly overpar.)	Practical scale (moderate width)
Pre(f_1) Post(f_3) Processing	$\approx$ Fixed (barely change)	Fixed (pre-learned or random)	Learned (adapt to data)
Features $\mathbf{x}_d$	$\approx$ Fixed	Fixed	Learned
Weights $\mathbf{W}$	Barely move	Full GD	Full GD
Rate matrix $\mathbf{R}_t$	$\approx$ Fixed ($\approx \mathbf{K}_0$)	Changes (via $\mathbf{W}, \mathbf{z}_d, \mathbf{g}_d$)	Varies
Linearized?	Yes	No (full nonlinear f_3)	No
Suboptimal minima avoided	Proven (NTK theory)	Proven (Part I)	Empirically yes; theory open
Convergence Rate	Single collective rate $\lambda_{\min}(\mathbf{K})$	Per-point rate α_d	Converges empirically
Generalization	Worse (random features)	Unknown	Best (learned features)

17.2 Equal $\hat{\lambda}_i$

This section restates and proves Theorem 2.7.1.

Theorem 2.7.1 (Equal $\hat{\lambda}_i$). *If $\forall i \hat{\lambda}_i = \hat{\lambda}$, then each data point d 's error $\mathcal{E}_d$ decays at its own rate $r_d = \alpha_d \hat{\lambda}$.*

Proof.

$$\begin{aligned}
r_{dd'} &:= (\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'}) && \text{(Definition)} \\
&= (\mathbf{g}_d^T \mathbf{g}_{d'}) \sum_j \hat{\lambda}_j \hat{v}_{dj} \hat{v}_{d'j} && \text{(Gram matrix: } \mathbf{x}_d^T \mathbf{x}_{d'} = [\hat{\mathbf{V}} \hat{\mathbf{\Lambda}} \hat{\mathbf{V}}^T]_{dd'}) \\
&= (\mathbf{g}_d^T \mathbf{g}_{d'}) \hat{\lambda} \sum_j \hat{v}_{dj} \hat{v}_{d'j} && \text{(All } \hat{\lambda}_j = \hat{\lambda}) \\
&= (\mathbf{g}_d^T \mathbf{g}_{d'}) \hat{\lambda} \delta_{dd'} && \text{(Orthonormality of } \hat{\mathbf{V}}) \\
&= \alpha_d \hat{\lambda} \delta_{dd'} && \text{(Diagonal: } \mathbf{g}_d^T \mathbf{g}_d = \alpha_d) \\
\partial e_d &= 2\eta \sum_{d'} r_{dd'} e_{d'} && \text{(Theorem 2.2)} \\
&= 2\eta \alpha_d \hat{\lambda} e_d. \\
\partial \mathcal{E}_d &= 4\eta \alpha_d \hat{\lambda} \mathcal{E}_d.
\end{aligned}$$

□

17.3 Ignoring Stalled Points

It is well known that if the error landscape gets flat then the gradient descent stalls. We show that this occurs on a point-by-point basis. When the gradient $\alpha_d := \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2 \rightarrow 0$, progress on its error $\mathcal{E}_d := (f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*)^2$ does stall. This progress comes in two forms: (i) on its own, $\mathcal{E}_d$ decays at rate $\Theta(\alpha_d)$ (since $\|\mathbf{x}_d\| = 1$), which clearly stalls with α_d ; and (ii) noise/coupling from other data points, which the following lemma shows also stalls, proportionally with $\sqrt{\alpha_d}$. Oddly when $\alpha_d \ll 1$, then $\sqrt{\alpha_d} \gg \alpha_d$. Moreover, other data points d' with reasonable gradients are not significantly affected by these stalled points. Hence, we partition data points into frozen (F) and active ($\bar{F}$). Frozen errors change negligibly; active errors evolve as if frozen points don't exist. The remaining analysis assumes all points are active.

Theorem 2.7.2 (Ignoring Stalled Points).

Points whose sensitivity α_d is too small can be ignored. Their error will not degrade quickly but they do not disturb the other points.

More formally: Partition data points into *flat* $F := \{d : \alpha_d \leq \alpha_F\}$ and *active* $A := \{d : \alpha_d \geq \alpha_A\}$, with $\alpha_F \ll (\frac{1}{\xi D})^2 \alpha_A$ and no points between.

($\alpha_F \ll \alpha_A$ when in the near-orthogonal regime with $\xi = O(1/D)$),

Here $\|\mathbf{x}_d\|^2 = 1$ and $|\mathbf{x}_d^T \mathbf{x}_{d'}| \leq \xi$ for $d \neq d'$.

Flat points barely move: For every $d \in F$,

$$|\partial e_d| \leq 2\eta \sqrt{\alpha_F} \left(\underbrace{\sqrt{\alpha_F} e_{\max}(t)}_{\text{self-drive vanishes with } \alpha_F} + \underbrace{\xi \sqrt{D \alpha_{\text{avg}} \mathcal{E}(t)}}_{\text{cross-coupling vanishes with } \sqrt{\alpha_F}} \right)$$

Active points barely feel F : Let $\partial e_{d'}|_A$ be the update ignoring flat points. Then

$$\frac{|\partial e_{d'} - \partial e_{d'}|_A|}{|\partial e_{d'}|} \ll 1$$

Proof.

Key bound. For any pair d, d' , Cauchy–Schwarz on $\mathbf{g}_d^T \mathbf{g}_{d'}$ gives

$$|r_{dd'}| = |(\mathbf{g}_d^T \mathbf{g}_{d'}) (\mathbf{x}_d^T \mathbf{x}_{d'})| \leq \|\mathbf{g}_d\| \|\mathbf{g}_{d'}\| |\mathbf{x}_d^T \mathbf{x}_{d'}| = \sqrt{\alpha_d \alpha_{d'}} |\mathbf{x}_d^T \mathbf{x}_{d'}|.$$

Part 1 (flat points barely move). Fix $d \in F$.

$$\begin{aligned} |\partial e_d| &= \left| 2\eta \sum_{d'} r_{dd'} e_{d'} \right| && \text{(per-point dynamics)} \\ &\leq 2\eta \sum_{d'} \sqrt{\alpha_d \alpha_{d'}} |\mathbf{x}_d^T \mathbf{x}_{d'}| |e_{d'}| && \text{(key bound + triangle ineq.)} \\ &= 2\eta \sqrt{\alpha_d} \underbrace{\sum_{d'} \sqrt{\alpha_{d'}} |\mathbf{x}_d^T \mathbf{x}_{d'}|}_{\Phi_d} |e_d|. && \text{(factor out } \sqrt{\alpha_d}) \end{aligned}$$

Split Φ_d into self ($d'=d$) and cross ($d' \neq d$):

$$\begin{aligned} \Phi_d &= \underbrace{\sqrt{\alpha_d} \|\mathbf{x}_d\|^2}_{=1} |e_d| + \sum_{d' \neq d} \sqrt{\alpha_{d'}} \underbrace{|\mathbf{x}_d^T \mathbf{x}_{d'}|}_{\leq \xi} |e_{d'}| \\ &\leq \sqrt{\alpha_F} e_{\max}(t) + \xi \sqrt{D \alpha_{\text{avg}}} \mathcal{E}(t), \end{aligned}$$

where the second term uses Cauchy–Schwarz: $\sum_{d'} \sqrt{\alpha_{d'}} |e_{d'}| \leq \sqrt{\sum \alpha_{d'}} \sqrt{\sum e_{d'}^2} = \sqrt{D \alpha_{\text{avg}}} \mathcal{E}$. Since $\sqrt{\alpha_d} \leq \sqrt{\alpha_F}$: $|\partial e_d| \leq 2\eta \sqrt{\alpha_F} \Phi_d \leq 2\eta \sqrt{\alpha_F} (\sqrt{\alpha_F} e_{\max} + \xi \sqrt{D \alpha_{\text{avg}}} \mathcal{E})$.

Part 2 (active points barely feel F). Fix $d' \in A$. The perturbation from flat points is

$$\begin{aligned} |\partial e_{d'} - \partial e_{d'}|_A| &= \left| 2\eta \sum_{d \in F} r_{d'd} e_d \right| && \text{(linearity in the sum)} \\ &\leq 2\eta \sum_{d \in F} \sqrt{\alpha_{d'} \alpha_d} \underbrace{|\mathbf{x}_{d'}^T \mathbf{x}_d|}_{\leq \xi} |e_d| && \text{(key bound; } d \neq d') \\ &\leq 2\eta \sqrt{\alpha_{d'}} \sqrt{\alpha_F} \xi \sum_{d \in F} |e_d|. && (\alpha_d \leq \alpha_F \text{ for } d \in F) \end{aligned}$$

The self-drive of d' is $|2\eta r_{d'd'} e_{d'}| = 2\eta \alpha_{d'} |e_{d'}|$. Dividing perturbation by self-drive:

$$\frac{|\text{perturbation}|}{|\text{self-drive}|} \leq \sqrt{\frac{\alpha_F}{\alpha_{d'}}} \xi \frac{\sum_{d \in F} |e_d|}{|e_{d'}|} \leq \sqrt{\frac{\alpha_F}{\alpha_A}} \xi D \frac{e_{\max}(t)}{|e_{d'}|}.$$

Since the self-drive lower-bounds $|\partial e_{d'}|$ in the near-orthogonal regime (cross terms within A are subdominant), we get $|\partial e_{d'} - \partial e_{d'}|_A|/|\partial e_{d'}| \ll 1$ whenever $\alpha_F \ll (1/\xi D)^2 \alpha_A$. $\square$

17.4 Worst case Weights $\mathbf{W}$ and Access $\mathbf{z} \in \mathbb{R}^{M'}$

This section discusses what happens in unlikely worst-case scenarios. These are illustrations of the theory working, not exceptions to it.

Theorem 2.7.3 (Worst case Weights $\mathbf{W}$). *Not a problem.*

$\mathbf{W} = \mathbf{0}$: To begin, what happens in the unlikely case that the weights are initialized to zero, $\mathbf{W} = \mathbf{0}$? The pre- f_3 values $\mathbf{z}_d = \mathbf{W} \mathbf{x}_d \in \mathbb{R}^{M'}$ will also be zero. But by assumption, f_3 is sensitive everywhere. Hence $\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d) = \nabla_{\mathbf{z}} f_3(\mathbf{0})$ is a typical gradient. Since all

$\mathbf{g}_d$ are identical, $\mathbf{R} = \bar{\alpha} \mathbf{X}^T \mathbf{X}$ with $\bar{\alpha} > 0$, so the eigenvalues λ_i remain positive. The weights are changed in the first iteration

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \eta \nabla_{\mathbf{W}} \mathcal{E} = \mathbf{0} - \eta [\nabla_{\mathbf{Z}} \mathcal{E}] \mathbf{X}^T \in \mathbb{R}^{M' \times M}.$$

This produces a rank-1 update to $\mathbf{W}$, but the first iteration still moves $\mathbf{W}$ away from zero and self-correction begins.

Extreme Window Mismatch: Here is an odd scenario in which $\mathbf{W}$ maps data into a subspace where f_3 is insensitive. Specifically, for any fixed weights $\mathbf{W}$, $\mathbf{z}_d = \mathbf{W} \mathbf{x}_d \in \mathbb{R}^{M'}$ sends the signal from the data $\mathbf{x}_d$ to the pre- f_3 values $\mathbf{z}_d$. The values of $\mathbf{x}_d \in \mathbb{R}^M$ that are actually considered span $\hat{D} := \text{Rank}(\mathbf{X}) \leq \min(D, M)$ dimensions, and hence the same for $\mathbf{z}_d \in \mathbb{R}^{M'}$. Suppose the window into f_3 is wide with $M' > 2\hat{D}$, so that the full dimension of $\mathbf{z}$ is not touched. To make this bottleneck concrete, split these M' hidden nodes into $\mathbf{z} = (\mathbf{z}^{(1)}, \mathbf{z}^{(2)})$, where $\mathbf{z}^{(1)} \in \mathbb{R}^{\hat{D}}$ are the values that $\mathbf{x}_d$ changes and $\mathbf{z}_d^{(2)}$ is some constant (which may or may not be zero). Normally this is not a problem, but we are considering adversarial situations.

We might have assumed that f_3 is sensitive to all of its entries $z_{m'}$ for $m' \in [M']$. But really we only assumed that $\|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2 = \alpha_d$, which only states that there is some direction in which we can change $\mathbf{z}_d$ that will produce this slope. It could well be that f_3 is sensitive to $\mathbf{z}^{(2)}$ but constant with respect to $\mathbf{z}^{(1)}$. This gives zeros within the first $\hat{D}$ rows of $\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d) \in \mathbb{R}^{M'}$. Following the gradient chain,

$$\mathbf{G} = [\mathbf{g}_d \mid d \in [D]] \in \mathbb{R}^{M' \times D}, \quad \nabla_{\mathbf{Z}} \mathcal{E} = 2 \sum_i e_{\|i} \mathbf{G} \text{Diag}(\mathbf{v}_i) \in \mathbb{R}^{M' \times D},$$

$$\nabla_{\mathbf{W}} \mathcal{E} = [\nabla_{\mathbf{Z}} \mathcal{E}], \mathbf{X}^T \in \mathbb{R}^{M' \times M}, \quad \Delta \mathbf{W} = \eta \nabla_{\mathbf{W}} \mathcal{E}, \quad \Delta \mathbf{z}_d = \Delta \mathbf{W} \mathbf{x}_d \in \mathbb{R}^{M'}.$$

Since $\mathbf{g}_d$ is zero in the $\mathbf{z}^{(1)}$ rows, those rows of $\mathbf{W}$ receive no gradient update. Hence, this first part $\mathbf{z}^{(1)}$ does not change during this gradient descent update. In fact, if f_3 continues to be insensitive to its first $\hat{D}$ entries, then these rows of $\mathbf{W}$ and of $\mathbf{z}_d$ will never change during gradient descent.

The fear, however, is that $\mathbf{z}^{(1)}$ is the part of the pre- f_3 signal that we care about. Recall $\mathbf{z}^{(1)} \in \mathbb{R}^{\hat{D}}$ are the values that change when we switch from one data point $\mathbf{x}_d$ to another. This makes it look like learning never occurs. Luckily, it does.

Let us now focus on the remaining rows. Recall that by hypothesis a very unusual adversary set up $\mathbf{W}^{(2)}$ to be such that $\mathbf{z}^{(2)} = \mathbf{W}^{(2)} \mathbf{x}_d$ is the same vector for each data point d . This is similar to our previous $\mathbf{W} = \mathbf{0}$ case. The very first iteration moves these rows of $\mathbf{W}$ away from this adversarial case and self-correction begins.

These are not counterexamples to Gradient Descent Theorems. In both adversarial cases above, Theorem 2 guarantees that *every* iteration, including this first one, reduces the error geometrically. Because in this first iteration all $\mathbf{g}_d$ are identical, the system can't distinguish between data points and hence treats them all the same. However, the first gradient update breaks this adversarial symmetry.

17.5 Second-Order Effects

Theorem 2.7.4 (Second-Order Effects).

The second order effects in gradient descent can be bounded if the step size η is sufficiently small.

Theorem 2 uses the first-order approximation, considering the local slope of f_3 but not its curvature. This section considers the higher-order terms.

$$\partial e_d = \underbrace{\mathbf{g}_d^T \partial \mathbf{z}_d}_{\text{1st order}} + \underbrace{\frac{1}{2} \partial \mathbf{z}_d^T H_d \partial \mathbf{z}_d}_{\text{2nd order}} + O(\|\partial \mathbf{z}_d\|^3), \quad (1)$$

where for simplicity we consider $e_d = f_3(\mathbf{z}_d) - y_d^*$ instead of $\mathcal{E}_{\|i}$, $\mathbf{g}_d = \nabla_{\mathbf{z}} f_3(\mathbf{z}_d) \in \mathbb{R}^{M'}$ is the local gradient of f_3 and $\alpha_d = \|\mathbf{g}_d\|^2$ its squared slope, $H_d = \nabla_{\mathbf{z}}^2 f_3(\mathbf{z}_d) \in \mathbb{R}^{M' \times M'}$ is the Hessian, and $\kappa_d = \|H_d\|$ is its curvature. The first-order term is $O(\eta)$ and the second-order term is $O(\eta^2)$.

Comparing their magnitudes: $\frac{|\text{2nd order}|}{|\text{1st order}|} \lesssim \frac{\frac{1}{2} \|H_d\| \|\partial \mathbf{z}_d\|^2}{\|\mathbf{g}_d\| \|\partial \mathbf{z}_d\|} \approx \frac{\frac{1}{2} \kappa_d (\eta^2 \alpha_d)}{(\eta \alpha_d)} = \frac{1}{2} \eta \kappa_d$. The main theorems, which consider only the first-order approximation, are therefore correct provided the step size satisfies $\eta \ll \frac{1}{\kappa_d}$.

Bounded Hessian for standard activation networks. For all standard activations used in deep learning practice, the bounded-Hessian condition holds. Suppose f_3 is a deep network of the form

$$f_3(\mathbf{z}) = \sigma_{L_3}(\mathbf{W}_{L_3} \sigma_{L_3-1}(\mathbf{W}_{L_3-1} \cdots \mathbf{W}_{L_1+1} \sigma_{L_1}(\mathbf{z}) \cdots)),$$

where each σ_l is applied elementwise.

- **Piecewise-linear activations (ReLU, Leaky ReLU, etc.):** $H_d = \mathbf{0}$ within each linear region (except at the finitely many transition points). Hence the first-order analysis is exact — the second-order term vanishes identically.
- **Smooth saturating activations (sigmoid, tanh, GELU, softplus):** These activations behave like the slowly-growing half of e^x : both σ'_l and σ''_l are globally bounded. If the weight matrices satisfy $\|\mathbf{W}_l\|_{\text{op}} \leq \omega$, the chain rule gives $\|H_d\| \leq \kappa$ for some constant κ depending on ω , L , and the activation bounds. The condition $\eta \ll 1/\kappa$ is therefore a mild constraint on the step size.
- **Unbounded-curvature activations:** High-degree polynomial activations (occasionally used in specialized architectures, e.g. physics-informed networks) can have unbounded σ'' , and the condition may fail without additional constraints on the pre-activation range.

17.6 Generalizing to $\langle \alpha, \beta \rangle$ -PL Sensitivity

Theorem 2.7.5 (Generalizing to $\langle \alpha, \beta \rangle$ -PL Sensitivity). *An easy adjustment in the proof.*

The theorems so far require f_3 to be *sensitive* at each data point: $\alpha_d = \|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2 \geq \alpha > 0$. This is a strong pointwise condition — it demands that the gradient of f_3 never becomes small regardless of the current error. We now relax this to a *Polyak–Łojasiewicz (PL) type* condition that permits the gradient to shrink as the error shrinks, while still guaranteeing convergence.

Definition 1 ($\langle \alpha, \beta \rangle$ -PL sensitivity). *We say f_3 is $\langle \alpha, \beta \rangle$ -PL sensitive at $\mathbf{z}_d$ if*

$$\|\nabla_{\mathbf{z}} f_3(\mathbf{z}_d)\|^2 \geq \alpha |e_d|^{\beta-1},$$

where $e_d = f_3(\mathbf{z}_d) - y_d^*$, $\alpha > 0$, and $\beta \geq 1$.

For $\beta = 1$ this reduces to the original constant-sensitivity condition $\alpha_d \geq \alpha$. For $\beta > 1$ the bound on α_d weakens as $|e_d| \rightarrow 0$, which is natural: a network that nearly fits the data may have a small gradient at that data point.

The proof goes through. The key step in the proof of Theorem 1 is $|\partial e_d| \approx 2\eta \alpha_d |e_d|$. Under $\langle \alpha, \beta \rangle$ -PL sensitivity, this becomes $|\partial e_d| \geq 2\eta \alpha |e_d|^\beta$. The rest of the proof is unchanged: the update is a contraction on $|e_d|$ at every step, with the same spectral structure.

Convergence rate. For $\beta = 1$ we get geometric decay. For $\beta > 1$ the discrete recursion $|e_d(t+1)| \leq |e_d(t)|(1 - 2\eta \alpha |e_d(t)|^{\beta-1})$ is well approximated by the ODE $\dot{e} = -c e^\beta$ (with $c = 2\eta \alpha$). The number of iterations needed is given by $T = \Theta\left(\frac{1}{\eta \alpha} \frac{1}{\beta-1} \left(\frac{1}{\epsilon}\right)^{\beta-1}\right)$ for $\beta > 1$ compared with $\Theta\left(\frac{1}{\eta \alpha} \log \frac{|e_{d0}|}{\epsilon}\right)$ for $\beta = 1$. Note the time to reduce the original error e_{d0} to one was $\log e_{d0}$ and is now so fast that it is not even included. The time to reduce the error from one to ϵ , however, has increased from logarithmic $\log \frac{1}{\epsilon}$ to polynomial $\left(\frac{1}{\epsilon}\right)^{\beta-1}$.

No Changes. There are no other changes in any of the results.

17.7 Removing Redundant Rows

Theorem 2.7.6 (Removing Redundant Rows).

A common intuition suggests that overparameterization improves optimization by expanding the basins of attraction and creating additional paths to global minima. We prove that this is not the case by proving that any redundant ($M > \hat{D}$) network is a “shadow” of the non-redundant ($M = \hat{D}$) case—same critical points, same basins, same convergence rates.

No Basin Expansion No Extra Paths: Practice has found that having wider and wider neural networks ($M > D$) helps. The standard hand-wavy overparameterization narrative from the literature suggests that such redundancy gives “Basin Expansion” increasing the “number of paths” to the global minimum. This section contradicts this. There are no functional gains, no additional optimization paths, no basin expansion. The effective search space is identical to the $M = k$ case.

More Likely Data Independence: Extra width certainly gives more chance for the latter data representations to be linearly independent. With $M \geq D$, having $k < D$ is not the intention. This extra width is fed into a nonlinear activation layer like sigmoid whose intention is to mix things up enough to make the data point features be linearly independent, $k = D$. (See Theorem 2.6.1.) Our matrix $\mathbf{X}$ is defined to be after this layer. Hence, if $k < D$, then this intention failed.

Extra Width Does not Help: This section proves that in this architecture once the dependencies are fixed, having wider width has no effect on the network. The reason $M > k$ is useless is that the weight component $\mathbf{W}_\perp$ orthogonal to the data has no effect on the output signal. It does not change during gradient descent and it does not matter what value it has. We prove this redundant network $\mathbf{N}$ is just a “shadow” of a non-redundant network $\tilde{\mathbf{N}}$. Their iterations are coupled: the quick convergence of one means the same for the other.

Formal Statement: Given a network $\mathbf{N}$ with post- f_1 data representation $\mathbf{X} \in \mathbb{R}^{M \times D}$ of rank k and $M \geq k$, we define a mirror network $\tilde{\mathbf{N}}$ by replacing $\mathbf{X}$ with $\tilde{\mathbf{X}} \in \mathbb{R}^{k \times D}$.

Targets $y^* \in \mathbb{R}^D$, f_3 , and the resulting error do not change. The gradient descent of the new network mirrors that of the old.

Proof.

Spectral Decomposition of the Data: Compute the SVD of the data matrix $\mathbf{X}$ to identify the “Active” subspace:

$$\mathbf{X} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$$

$\mathbf{V} \in \mathbb{R}^{D \times k}$: The principal **right-singular vectors**. These form an orthonormal basis for the row space of $\mathbf{X}$. (They are the eigenvectors of $\mathbf{X}^T \mathbf{X}$.)

$\mathbf{\Sigma} \in \mathbb{R}^{k \times k}$: The diagonal matrix of **singular values** σ_i . Note that the eigenvalues of the data matrices are $\lambda_i = \sigma_i^2$.

$\mathbf{U} \in \mathbb{R}^{M \times k}$: The principal **left-singular vectors**. These form an orthonormal basis for the column space (range) of $\mathbf{X}$. (They are the eigenvectors of $\mathbf{X} \mathbf{X}^T$.)

It is important to distinguish that with $k < M$:

- $\mathbf{U}^T \mathbf{U} = \mathbf{I}_k \in \mathbb{R}^{k \times k}$ — $\mathbf{U}$ has orthonormal columns.
- However, $\mathbf{U} \mathbf{U}^T \in \mathbb{R}^{M \times M}$ projects any vector onto the k -dimensional subspace spanned by the data and is *not* the identity matrix.

The “Active” Subspace: The columns of $\mathbf{U}$ represent the only directions in the M -dimensional feature space in which the weights $\mathbf{W}$ can actually change the pre-activations $\mathbf{z}_d$.

Decomposition of the Weights $\mathbf{W} \in \mathbb{R}^{M' \times M}$:

$\mathbf{W}_{\parallel} = \mathbf{W} \mathbf{U} \mathbf{U}^T$ (**Active Part**): Every row lies in the column space of $\mathbf{U}$ (i.e., in $\text{span}(\mathbf{X})$). This is the only part that affects the output.

$\mathbf{W}_{\perp} = \mathbf{W}(\mathbf{I} - \mathbf{U} \mathbf{U}^T)$ (**Ghost Part**): Every row is orthogonal to $\text{span}(\mathbf{X})$, meaning $\mathbf{W}_{\perp} \mathbf{x}_d = 0$ for every data point. When multiplied by the data matrix, the ghost part vanishes entirely:

$$\mathbf{W}_{\perp} \mathbf{X} = \mathbf{W}(\mathbf{I} - \mathbf{U} \mathbf{U}^T) \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T = \mathbf{0},$$

since $(\mathbf{I} - \mathbf{U} \mathbf{U}^T) \mathbf{U} = \mathbf{U} - \mathbf{U}(\mathbf{U}^T \mathbf{U}) = \mathbf{U} - \mathbf{U} = \mathbf{0}$.

$\mathbf{W}_{\perp}$ is Fixed: At every time step t : $\mathbf{W}(t) = \mathbf{W}_{\parallel}(t) + \mathbf{W}_{\perp}$. The gradient is $\nabla_{\mathbf{W}} \mathcal{E} = (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{X}^T = (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{V} \mathbf{\Sigma} \mathbf{U}^T$. Every row of $\nabla_{\mathbf{W}} \mathcal{E}$ is in the column space of $\mathbf{U}$, so $\nabla_{\mathbf{W}} \mathcal{E} \cdot (\mathbf{I} - \mathbf{U} \mathbf{U}^T) = \mathbf{0}$: the gradient has no perpendicular component. Under gradient descent, $\mathbf{W}_{\perp}$ receives no update and never changes. It is not a problem that it does not change, because it has no effect anyway.

Mirror Network: The morphism between the redundant network $\mathbf{N}$ and the independent mirror $\tilde{\mathbf{N}}$ is as follows. Note that the number of features in the data representation $\mathbf{X}$ decreases from M to k , making these rows linearly independent.

Component	Original $\mathbf{N}$	Mirror $\tilde{\mathbf{N}}$	Mapping
Data	$\mathbf{X} \in \mathbb{R}^{M \times D}$	$\tilde{\mathbf{X}} \in \mathbb{R}^{k \times D}$	$\tilde{\mathbf{X}} = \Sigma \mathbf{V}^T, \mathbf{X} = \mathbf{U} \tilde{\mathbf{X}}$
Features	Redundant ($M > k$)	Independent ($M = k$)	SVD removes null rows
Weights	$\mathbf{W} \in \mathbb{R}^{M' \times M}$	$\tilde{\mathbf{W}} \in \mathbb{R}^{M' \times k}$	$\mathbf{W}_{\parallel} = \tilde{\mathbf{W}} \mathbf{U}^T$
Null space	$\mathbf{W}_{\perp} = \mathbf{W}(\mathbf{I} - \mathbf{U}\mathbf{U}^T)$	None	$\mathbf{W}_{\perp}$ frozen, no effect
Pre-activations	$\mathbf{Z} = \mathbf{W}\mathbf{X}$	$\tilde{\mathbf{Z}} = \tilde{\mathbf{W}} \tilde{\mathbf{X}}$	$\mathbf{Z} = \tilde{\mathbf{Z}}$ (identical)
Error	Since $\mathbf{Z}(t) = \tilde{\mathbf{Z}}(t)$, the network's output and error $\tilde{\mathcal{E}}(\tilde{\mathbf{W}}) = \mathcal{E}(\mathbf{W})$ are the same.		
Gradient	$\nabla_{\mathbf{W}} \mathcal{E} = (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{X}^T$	$\nabla_{\tilde{\mathbf{W}}} \mathcal{E} = (\nabla_{\mathbf{Z}} \mathcal{E}) \tilde{\mathbf{X}}^T$	$\nabla_{\mathbf{W}} \mathcal{E} = \nabla_{\tilde{\mathbf{W}}} \mathcal{E} \cdot \mathbf{U}^T$
Eigenvalues	$\lambda_i(\mathbf{X}\mathbf{X}^T)$	$\lambda_i(\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T)$	Identical

Coupled: This morphism between $\mathbf{N}$ and $\tilde{\mathbf{N}}$ can be set up for time zero. What remains to complete the induction is to prove that each step of gradient descent maintains it.

$$\begin{aligned}
\mathbf{Z}(t) &= \mathbf{W}(t) \cdot \mathbf{X} && \text{(Definition)} \\
&= [\mathbf{W}_{\parallel}(t) + \mathbf{W}_{\perp}] \cdot \mathbf{X} && \text{(Decompose)} \\
&= \mathbf{W}_{\parallel}(t) \cdot \mathbf{X} && \text{(Ghost vanishes: } \mathbf{W}_{\perp} \mathbf{X} = \mathbf{0}) \\
&= [\tilde{\mathbf{W}}(t) \cdot \mathbf{U}^T] \cdot [\mathbf{U} \Sigma \mathbf{V}^T] && \text{(Mapping and SVD)} \\
&= \tilde{\mathbf{W}}(t) \cdot \Sigma \mathbf{V}^T && (\mathbf{U}^T \mathbf{U} = \mathbf{I}_k) \\
&= \tilde{\mathbf{W}}(t) \cdot \tilde{\mathbf{X}} = \tilde{\mathbf{Z}}(t) && \text{(Definition)}
\end{aligned}$$

Mirrored Pre-activations:

Mirrored Gradient: Both gradients follow from the standard formula $\nabla_{\mathbf{W}} \mathcal{E} = (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{X}^T$:

$$\begin{aligned}
\nabla_{\mathbf{W}} \mathcal{E} &= (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{X}^T = (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{V} \Sigma \mathbf{U}^T \\
\nabla_{\tilde{\mathbf{W}}} \mathcal{E} &= (\nabla_{\mathbf{Z}} \mathcal{E}) \tilde{\mathbf{X}}^T = (\nabla_{\mathbf{Z}} \mathcal{E}) \mathbf{V} \Sigma
\end{aligned}$$

Comparing: $\nabla_{\mathbf{W}} \mathcal{E} = \nabla_{\tilde{\mathbf{W}}} \mathcal{E} \cdot \mathbf{U}^T$ (equivalently, $\nabla_{\tilde{\mathbf{W}}} \mathcal{E} = \nabla_{\mathbf{W}} \mathcal{E} \cdot \mathbf{U}$).

$$\begin{aligned}
\mathbf{W}(t+1) &= \mathbf{W}(t) - \eta \nabla_{\mathbf{W}} \mathcal{E} && \text{(Gradient descent)} \\
&= [\tilde{\mathbf{W}}(t) \mathbf{U}^T + \mathbf{W}_{\perp}] - \eta [\nabla_{\tilde{\mathbf{W}}} \mathcal{E} \cdot \mathbf{U}^T] && \text{(Mapping and mirrored gradient)} \\
&= [\tilde{\mathbf{W}}(t) - \eta \nabla_{\tilde{\mathbf{W}}} \mathcal{E}] \cdot \mathbf{U}^T + \mathbf{W}_{\perp} \\
&= \tilde{\mathbf{W}}(t+1) \cdot \mathbf{U}^T + \mathbf{W}_{\perp} && \text{(Gradient descent in } \tilde{\mathbf{N}})
\end{aligned}$$

Induction Step:

Hence by induction, the two networks stay mirrored for all t . □

17.8 Ghost Error

Theorem 2.7.7 (Ghost Error Decays).

Simple models of rate-matrix rotation and stochastic gradient descent show that the ghost error decays to zero.

Proof. While the rate matrix $\mathbf{R}$ is approximately constant, Theorem 1.1 shows the non-ghost error decays, while the $k := D - MM'$ -dimensional ghost error persists.

Rotating the Rate Matrix: As gradient descent proceeds, however, $\mathbf{R}_t$ changes, rotating the definition of the ghost subspace. Each iteration, the non-ghost component under the current $\mathbf{R}_t$ decays. If $\mathbf{R}_t$ keeps changing, different amounts of the total error may eventually decay. Note that different initial weights $\mathbf{W}_0$ produce different $\mathbf{R}_0$ with different null spaces, so the ghost error depends on the starting point: different initializations lead to different valleys with different final error levels.

Phases with Different Rate Matrices: Given any error $\mathcal{E}_d$ and a random matrix $\mathbf{R}_t$ with rank deficit $k = D - \text{Rank}(\mathbf{R}_t)$, Theorem 2.1 shows that the error is partitioned evenly among the D channels, so the expected amount of ghost error is $\frac{k}{D}\mathcal{E}_d$. After $\Theta\left(\frac{1}{\eta \alpha_{\min}} \log \frac{\mathcal{E}_d}{\epsilon}\right)$ iterations with this rate matrix, all of its non-ghost error has decayed leaving only $\frac{k}{D}\mathcal{E}_d$ of the error. If this is repeated for T such independent phases, the error is down to $\left(\frac{k}{D}\right)^T \mathcal{E}_d$.

Between Phases: Consider any current error vector $\mathbf{e} \in \mathbb{R}^D$ giving $e_d := f_{\mathbf{W}}(\mathbf{a}_d) - y_d^*$. When gradient descent updates $\mathbf{W}$, the rate matrix changes from $\mathbf{R}_1$ to $\mathbf{R}_2$. The error vector $\mathbf{e}_t$ does not jump; the next step uses $\mathbf{e}_{t+1} = (\mathbf{I} - 2\eta\mathbf{R}_2)\mathbf{e}_t$ with the new rate matrix. What changes is the decomposition into ghost and non-ghost.

Formally, let P_1 and P_2 project onto $\ker(\mathbf{R}_1)$ and $\ker(\mathbf{R}_2)$, respectively. The old ghost component $P_1\mathbf{e}$ splits: $P_2P_1\mathbf{e}$ remains ghost under $\mathbf{R}_2$, while $(\mathbf{I} - P_2)P_1\mathbf{e}$ is now non-ghost and will decay in the next phase. The fraction of ghost error absorbed depends on the angle between the two null spaces.

Why $\mathbf{R}_t$ rotates: Even with fixed f_1 and f_3 , as $\mathbf{W}_t$ changes, $\mathbf{z}_d = \mathbf{W}_t\mathbf{x}_d$ changes, so $\mathbf{g}_d = \nabla_{\mathbf{z}}f_3(\mathbf{z}_d)$ rotates (for any nonlinear f_3). The only case where $\mathbf{R}$ is truly frozen is linear f_3 (where $\mathbf{g}_d$ is constant).

The catch: Ghost error generates zero gradient, so all null-space rotation is a *side effect* of the non-ghost gradient. As non-ghost error decays, this rotation slows ($\sum_{t=T}^{\infty} \|\Delta\mathbf{W}_t\| < \infty$), bounding the total rotation. Whether enough rotation occurs to absorb all ghost error before the non-ghost gradient vanishes is open (Open Problem 6).

Stochastic Gradient Descent: Stochastic gradient descent adds random noise to the weight matrix $\mathbf{W}$. This has the positive effect of rotating $\mathbf{R}$ but also has the negative effect of creating more overall error.

Let $\mathcal{E}_{Gt}$ and $\mathcal{E}_{Nt}$ be the amounts of ghost and non-ghost error at iteration t . Gradient descent will run for several iterations, decreasing the non-ghost error from $\mathcal{E}_{Nt}$ to $r\mathcal{E}_{Nt}$. Suppose noise is added to $\mathbf{W}$. Let k' denote the expected number of channels that switch between ghost and non-ghost. Of the ghost's k channels, $\frac{k'}{k}\mathcal{E}_{Gt}$ of the ghost error will become non-ghost and similarly $\frac{k'}{D}r\mathcal{E}_{Nt}$ of the non-ghost error will become ghost. Let b denote the expected amount of additional error added by the noise in $\mathbf{W}$. Being random error, it is split evenly among the D channels. Hence $\frac{k}{D}b$ will be added to the ghost error and $\frac{D'}{D}b$ to the non-ghost. This gives:

$$\begin{aligned}\mathcal{E}_{G,t+1} &= \mathcal{E}_{Gt} - \frac{k'}{k}\mathcal{E}_{Gt} + \frac{k'}{D}r\mathcal{E}_{Nt} + \frac{k}{D}b \\ \mathcal{E}_{N,t+1} &= r\mathcal{E}_{Nt} + \frac{k'}{k}\mathcal{E}_{Gt} - \frac{k'}{D}r\mathcal{E}_{Nt} + \frac{D'}{D}b\end{aligned}$$

The total error converges to $\left(\frac{1}{1-r} + \frac{k^2}{k'D}\right)b$. When enough channels switch ($k' = \Theta(k)$), ghost error leaks into non-ghost channels faster than new ghost error arrives, and the non-ghost component decays (Open Problem 6). $\square$

Part III

Insensitive Activations (Boolean) (NP-completeness, Algorithm)

Our table of results in Section 4 is split into two columns. Part II handled the left column, studying how quickly gradient descent converges with “Sensitive Activations (Real).” We now handle the right “Insensitive Activations (Boolean).” The problem is formulated as follows. We have the data representation $\mathbf{x}_d = f_1(\mathbf{a}_d)$ produced by the first stages f_1 of the neural network and we will assume that understanding the final stages f_3 is not the problem. Computing the output $f_3(z^*) = y^*$ from the pre- f_3 data representation $z^* \in \mathbb{R}^M$ is easy as is inverting it (non-deterministically) finding a $z^* \in \mathbb{R}^M$ with $f_3(z^*) = y^*$ for any required output $y^* \in \mathbb{R}$. The difference from the left column, however, is that there is no gradient. A small change in z^* either does not change or completely changes y^* . The task at hand is to find an optimizing weight matrix $\mathbf{W}$. This could be the decision problem of whether a zero-error solution exists, minimizing the sum error $\mathcal{E}$ problem $\min_{\mathbf{W}} \sum_d (f_3(\mathbf{W}\mathbf{x}_d) - y_d^*)^2$, or minimizing the max error $\min_{\mathbf{W}} \max_d (f_3(\mathbf{W}\mathbf{x}_d) - y_d^*)^2$.

The rows of the Section 4 table (for this section at least) are determined by the rank of the data representation $\mathbf{X}$. If the points, represented as $\mathbf{x}_d$ are linearly independent, solving for $\mathbf{W}$ is simple linear algebra. If dependent, we adjust Blum and Rivest [1992]’s 3-SAT NP-Complete reduction to fit into our framework. In between, (assuming the Exponential Time Hypothesis) this same result translates into a $2^{\Omega(\hat{k})}$ lower bound when $\mathbf{X}$ has rank deficit $\hat{k} = D - \hat{D}$. We give a $D^{O(M'\hat{k}^2)}$ algorithm for the min-max problem; it is polynomial for each fixed $\hat{k}$ and M' (the class XP), and does not match the lower bound.

Theorem 4 (Boolean f_3 : Rank Phase Transition to Hardness).

1. **Linearly Independent:** If the data representation $\mathbf{X}$ has full rank and for every d there exists z_d^* with $f_3(z_d^*) = y_d^*$, then a weight matrix $\mathbf{W}$ achieving zero training error exists.⁴⁸

Proof: $\mathbf{W} = \mathbf{Z}^* \mathbf{X}^+$, where $\mathbf{X}^+ = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T$ is the pseudoinverse (a left-inverse: $\mathbf{W}\mathbf{X} = \mathbf{Z}^* \mathbf{X}^+ \mathbf{X} = \mathbf{Z}^*$).

2. **NP-Completeness:** Consider training a neural network with M input nodes, one fully connected hidden layer to M' insensitive sigmoid/threshold nodes, to a single M' -ary AND output node. Deciding whether a zero-error $\mathbf{W}$ exists for a complicated $\mathbf{X}$ is NP-complete.

(For any $\hat{k} \geq 1$, $M' \geq 2$, and $M \geq n$ where n is the 3SAT instance size.)

3. **Lower Bound:** If the D training points have rank $\hat{D} = D - \hat{k}$, then under the Exponential Time Hypothesis, any algorithm requires $2^{\Omega(\hat{k})}$ time.
4. **XP Algorithm:** For $f_3(\mathbf{z}_d) = \hat{f}_3(\text{Threshold}(\mathbf{z}_d))$, for an arbitrary $\hat{f}_3 : \{0,1\}^{M'} \rightarrow \mathbb{R}$, there is an algorithm running in $D^{O(M'\hat{k}^2)}$ time solving $\min_{\mathbf{W}} \max_d (f_3(\mathbf{z}_d) - y_d^*)^2$.

⁴⁸**Not in Range:** Consider a sigmoid output $f_3(z) \in [0,1]$ and a target $y^* = 5$. The error $(f_3(z) - 5)^2$ is minimized when $f_3(z) \rightarrow 1$, requiring $z \rightarrow \infty$. The gradient asymptotically approaches zero as the weights grow; no finite critical point exists. **Constraint of Dependence:** If $\mathbf{x}_i = \sum_j c_j \mathbf{x}_j$, then $\mathbf{z}_i = \sum_j c_j \mathbf{z}_j$, so $f_3(\mathbf{z}_i)$ is fixed by the other points; if y_i^* does not match, zero error is impossible.

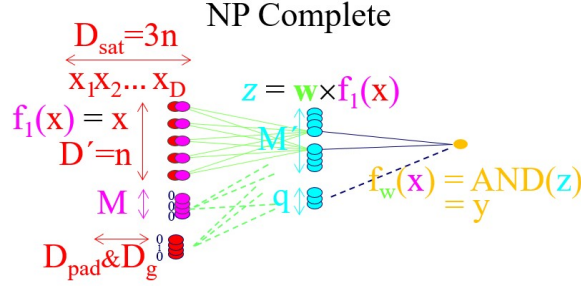


Figure 3: Blum–Rivest reduction embedded in our framework, with padding and gadget points.

18 NP-Completeness Reduction

Proof of Theorem 4.2, 4.3 binary-AND.

Overview: We start with the original Blum and Rivest [1992] reduction from an instance of 3SAT conjectured to require $2^{\Omega(n)}$ time to an instance of network weight training with $M = n$, $M' = 2$, and rank deficit $\hat{k} = D - \hat{D} = \Theta(n)$. We obtain the $2^{\Omega(\hat{k})}$ time lower bound simply by padding the data with independent points increasing D and $\hat{D}$ together keeping the rank deficit $\hat{k} = D - \hat{D}$ and the required time $2^{\Omega(\hat{k})}$ fixed. To make the result more general and to challenge the preconception that width helps, we further pad to increase M and M' arbitrarily. To obfuscate, we rotate the feature vectors hiding the $\hat{D}$ -dim subspace $\mathbf{X}$ within $\mathbb{R}^M$.

NP-Reduction: Blum and Rivest map an n -variable $m = n - 1$ clause instance of 3SAT to a neural network with n input nodes, one fully connected hidden layer with 2 threshold nodes, a single fixed AND output node, and $D = 2n + m + 1 = 3n$ training points. Note that $\hat{D} = \text{Rank}(\mathbf{X}) = n = \frac{1}{3}D$, given $D = 3n$ data vectors of length n .

Framework $\mathbf{f}_W(\mathbf{a}) = \mathbf{f}_3(\mathbf{W} \cdot \mathbf{f}_1(\mathbf{a}))$: See Figures 1 and 3.

- Let the fixed $f_1(\mathbf{a}) = \mathbf{X} \in \{0,1\}^n$ be the identity map so that our “feature vectors” $\mathbf{X} = [f_1(\mathbf{a}_d) | d \in [D]] \subset \mathbb{R}^n$ are just the training points themselves and $M = n$.
- $\mathbf{W} \in \mathbb{R}^{2 \times n}$: A matrix of trainable weights.
- Let the fixed $f_3 : \mathbb{R}^2 \rightarrow \mathbb{R}$ be two steep sigmoids/thresholds into a binary AND gate. In order to fulfill Part II’s requirement that f_3 possesses local sensitivity, it can be made to be differentiable with non-zero (though potentially small) gradients.

Proof for the $2^{\Omega(\hat{k})}$ Lower Bound: By the Exponential Time Hypothesis, our current $D_{\text{sat}} = 3n$ data points, $\hat{D}_{\text{sat}} = n$ rank, $\hat{k}_{\text{sat}} = 2n$ deficit network requires $2^{\Omega(n)} = 2^{\Omega(\hat{k}_{\text{sat}})}$ time to solve. As said, we pad the data with $D_{\text{pad}} = D_{\text{new}} - D_{\text{sat}}$ independent points increasing D and $\hat{D}$ together keeping $\hat{k} = D_{\text{new}} - \hat{D}_{\text{new}}$ fixed. This is done by extending the input dimension from n to $n + D_{\text{pad}}$ and defining the auxiliary points as unit vectors $\mathbf{1}_j$ in the new dimensions. By setting the target labels for these points to $y^* = 0$, we ensure they are trivially satisfied by setting the corresponding weights to zero. Since these new points are linearly independent of the original set and each other, the rank increases exactly by D_{pad} to $\hat{D}_{\text{new}} = \hat{D}_{\text{sat}} + D_{\text{pad}}$. The deficit $\hat{k}_{\text{new}} = D_{\text{new}} - \hat{D}_{\text{new}} = D_{\text{sat}} - \hat{D}_{\text{sat}} = \hat{k}_{\text{sat}}$ remains invariant, and the $2^{\Omega(\hat{k})}$ hardness

of the original 3SAT core is preserved within the larger dataset.

Note the time $2^{\Omega(k)}$ matches our Theorem 1 independent result for $\hat{k}=0$ and our Theorem 4.2 NP-complete result for $\hat{k}=\frac{2}{3}D$.

Making M Arbitrarily Large: Increase the number of inputs nodes in the neural network from n to M . Pad the corresponding elements of each training point with zero. These additional trainable weights going from $\mathbf{W} \in \mathbb{R}^{2 \times n}$ to $\mathbb{R}^{2 \times M}$ have no effect because they are all multiplied by these zeros.

Making M' Arbitrarily Large: To increase the number of trainable weights further from $\mathbf{W} \in \mathbb{R}^{2 \times M}$ to $\mathbb{R}^{M' \times M}$, split the two nodes $\mathbf{z}_1$ and $\mathbf{z}_2$ into $\{\mathbf{z}_{\langle 1, i \rangle} \mid i \in [\frac{1}{2}M']\}$ and $\{\mathbf{z}_{\langle 2, i \rangle} \mid i \in [\frac{1}{2}M']\}$. To keep the logic intact, the fixed binary AND gate $f_3 : \mathbb{R}^2 \rightarrow \mathbb{R}$ must evolve to $f_3(\mathbf{z}) = \text{AND}(\sum_{i \in [\frac{1}{2}M']} \mathbf{z}_{\langle 1, i \rangle}, \sum_{i \in [\frac{1}{2}M']} \mathbf{z}_{\langle 2, i \rangle})$.

Rotate $\mathbf{X}$: Though it does not hurt the statement of the theorem, having all of this structure in the data points is not aesthetically pleasing. We can simply rotate them away. Let $\mathbf{Q} \in \mathbb{R}^{M \times M}$ be an arbitrary rank- M matrix. Replacing the training points/feature vectors $\mathbf{X} = [f_1(\mathbf{a}_d) \mid d \in [D]]$ with $\mathbf{Q} \cdot \mathbf{X}$ and the trained weights $\mathbf{W}$ with $\mathbf{W} \cdot \mathbf{Q}^{-1}$ has no effect on the pre-activations $\mathbf{z} = \mathbf{W} \cdot f_1(\mathbf{a}) = (\mathbf{W} \cdot \mathbf{Q}^{-1}) \cdot (\mathbf{Q} \cdot f_1(\mathbf{a}))$. $\square$

19 Increasing the Arity M' of the AND

While this result formally fits into the framework, it is theoretically unsatisfying: two hidden nodes need to do all the heavy lifting while the input dimension scales. We now generalize the binary AND gate to a M' -ary AND gate with $M' \leq \frac{D}{4 \log D}$. We do this by “distracting” the extra hidden nodes with capacity-draining gadgets.

Proof of Theorem 4.2, 4.3, and large M' . We start with a $M'_{sat}=2$ neural network with M_{sat} input nodes, $M'_{sat}=2$ -ary & , and D_{sat} data points. We then scale the number of hidden nodes to M' by introducing $M'-2$ capacity-draining gadgets on a disjoint set of M_g input nodes. Our final neural network will have $M_{new}=M_{sat}+M_g$ input nodes, one fully connected hidden layer with M' threshold nodes, a single M' -ary AND output node, and $D_{new}=D_{sat}+D_g$ training points.

The Blum–Rivest Gadget: The core difficulty in training this network is “resource contention.” Consider the gadget subspace as a Boolean hypercube $\{0,1\}^{M_g}$ on M_g of the input variables, with the other M_{sat} variables set to zero. A single gadget consists of one training input labeled $y^*=0$ located at a specific corner of this hypercube, and M_g training inputs labeled $y^*=1$ located at the corner’s immediate neighbors.

Because the output node is an AND gate, all positive examples require every hidden node to output 1. Conversely, a negative example requires at least one hidden node to output 0. Therefore, a single hidden node must use its hyperplane to slice off exactly this $y^*=0$ corner without misclassifying the surrounding $y^*=1$ neighbors. We include $M'-2$ such disjoint gadgets to effectively lock the weights of $M'-2$ hidden nodes. This leaves exactly $M'_{sat}=2$ hidden nodes free to encode the logic of the 3SAT instance.

Embedding the SAT Instance: Blum and Rivest map an n -variable instance of 3SAT to the remaining $M_{sat}=n$ input variables using $D_{sat}=\mathcal{O}(n)$ training points. They prove that

training these remaining 2 nodes to fit the data is equivalent to solving the original 3SAT instance.

The Capacity Scaling: To ensure the gadgets do not interfere with one another, their centers must be placed at least a Hamming distance of 3 apart. The Gilbert–Varshamov bound guarantees we can pack this many centers into the hypercube provided $M' - 2 \leq \frac{2^{M_g+1}}{M_g^2}$. Solving for the required gadget dimension gives $M_g \approx \log_2 M'$. The total amount of gadget training data needed is $D_g = (M' - 2)(M_g + 1) \approx M' \log_2 M'$. The total amount of data will be about the same giving $M' \approx \frac{D_{new}}{\log D_{new}}$. All of these new data points are linearly dependent, making $\hat{D} \leq (1-o(1))D$ worse, but this can be restored by adding more linearly independent points. $\square$

20 Algorithm for the $\hat{k} = D - \hat{D}$ Rank Deficit Problem

Theorem 4.4 (XP Algorithm).

Consider training a neural network taking inputs $\mathbf{x}_d \in \mathbb{R}^M$, feeding a fully connected hidden layer $\mathbf{z}_d = \mathbf{W}\mathbf{x}_d \in \mathbb{R}^{M'}$, feeding thresholds $\hat{\mathbf{z}}_d = \text{Threshold}(\mathbf{z}_d)$, into an arbitrary output function $\hat{f}_3 : \{0,1\}^{M'} \rightarrow \mathbb{R}$. There is an algorithm running in $D^{O(M'\hat{k}^2)}$ time solving $\min_{\mathbf{W}} \max_d (f_3(\mathbf{W}\mathbf{x}_d) - y_d^*)^2$.

Surprisingly, this is fixed-parameter tractable, being hard only in $\hat{k}$ and M' , not in the data size D . Non-linearity is managed by enumerating over the $2^{M'}$ possible thresholds of the $\hat{k}$ dependent data points. We have no idea of the complexity of minimizing the error when no zero-error solution exists; it likely is NP-complete even for $\hat{k} = 1$.

Proof.

Binary Search for Max Error ϵ : We solve $\epsilon^* = \min_{\mathbf{W}} \max_d (f_3(\mathbf{W}\mathbf{x}_d) - y_d^*)^2$ by doing binary search on the value of ϵ^* . For each trial ϵ , we ask whether there exist weights $\mathbf{W}$ such that $\forall d (f_3(\mathbf{W}\mathbf{x}_d) - y_d^*)^2 \leq \epsilon$.

In contrast, solving $\min_{\mathbf{W}} \sum_d (f_3(\mathbf{W}\mathbf{x}_d) - y_d^*)^2$ seems to be much harder because even if this point-wise tradeoff were narrowed to deciding for each d , whether to force $(f_3(\mathbf{W}\mathbf{x}_d) - y_d^*)^2$ to be ϵ_1^* or to be ϵ_2^* , this still leaves 2^D combinations to try.

$\hat{D}$ Independent and $\hat{k}$ Dependent: Order the data points so that the first $\hat{D} = \text{Rank}(\mathbf{X})$ (denoted $\mathbf{X}_1$, $\mathbf{Z}_1$, and $\hat{\mathbf{Z}}_1$) are linearly independent. And the remaining $\hat{k} = D - \hat{D}$ (denoted $\mathbf{X}_2$, $\mathbf{Z}_2$, and $\hat{\mathbf{Z}}_2$) are dependent on these. We will abuse notation by writing $\forall d \in [\hat{D}]$ instead of $\forall d \in [1, \hat{D}]$ and $\forall d \in [\hat{k}]$ instead of $\forall d \in [\hat{D} + 1, D]$.

Dependency Matrix: Define $\mathbf{C} \in [-D, D]^{\hat{D} \times \hat{k}}$ by $\forall k \in [\hat{k}] \mathbf{x}_{\hat{D}+k} = \sum_{d=1}^{\hat{D}} c_{dk} \mathbf{x}_d$. This allows us to drop caring about the data points $\mathbf{x}_d$, the weight matrix $\mathbf{W}$, and the relation $\mathbf{z}_d = \mathbf{W}\mathbf{x}_d$. Instead, we only require that $\forall k \in [\hat{k}] \mathbf{z}_{\hat{D}+k} = \sum_{d=1}^{\hat{D}} c_{dk} \mathbf{z}_d$. We assume each c_{dk} is an integer in $[-D, D]$.⁴⁹

⁴⁹If $\mathbf{X}$ has rational entries (as in the Blum–Rivest construction with entries in $\{0, 1\}$), the dependency coefficients are rational, computable by Cramer’s rule on the $\hat{D} \times \hat{D}$ subsystem. Clearing the common denominator gives integers bounded by $D^{O(1)}$.

Enumerating All $\hat{\mathbf{Z}}_2$: We have $D^{O(M'\hat{k}^2)}$ time. Hence, we cannot consider each of the $2^{M'\hat{D}}$ post-threshold pre- $\hat{f}_3$ vectors $\hat{\mathbf{Z}}_1 \in \{0, 1\}^{M' \times \hat{D}}$ for the $\hat{D}$ independent data points. However, because there are only $2^{M'\hat{k}}$ combinations for the $\hat{k}$ dependent data points, we can and do enumerate over these. Given the current one $\hat{\mathbf{Z}}_2 \in \{0, 1\}^{M' \times \hat{k}}$, we accept iff

- $\forall d \in [\hat{k}] (f_3(\hat{\mathbf{z}}_d) - y_d^*)^2 \leq \epsilon$ (Correct output on the $\hat{k}$)
- $\exists \mathbf{Z}_1 \in \mathbb{R}^{M' \times \hat{D}}$: (Pre-threshold vectors for the $\hat{D}$)
 - $\forall k \in [\hat{k}] \hat{\mathbf{z}}_{\hat{D}+k} = \text{Threshold}(\sum_{d=1}^{\hat{D}} c_{dk} \mathbf{z}_d)$ (Correct Dependencies)
 - $\forall d \in [\hat{D}] (f_3(\mathbf{z}_d) - y_d^*)^2 \leq \epsilon$ (Correct output on the $\hat{D}$)

Disentangling the $\hat{D}$ Vectors $\mathbf{z}_d \in \mathbb{R}^{M'}$: Instead of worrying about the $\geq 2^{\hat{D}}$ dynamics between these $\hat{D}$ vectors, we want to independently find each vector $\mathbf{z}_d \in \mathbb{R}^{M'}$ one at a time. They, however, are linked via the sum $\sum_{d=1}^{\hat{D}} c_{dk} \mathbf{z}_d \in \mathbb{R}^{M'}$. The standard way to handle this is using dynamic programming by considering all of the possible sums $\sum_{d'=1}^d c_{d'k} \mathbf{z}_{d'}$ that can be made from the first d of the $\hat{D}$ vectors. The problem with this algorithm is that the entries $z_{m'd}$ are real numbers and this creates too many possible sums to consider. The fix to this bug is to replace the real numbers with integers.

Replace Reals with Integers: We need not search over $\mathbf{z}_d \in \mathbb{R}^{M'}$; it suffices to consider $\mathbf{z}_d \in [-L, L]^{M'}$ where $L = D^{O(\hat{k})}$. The key observation is that the $\hat{k}$ dependent constraints and sign constraints define a polyhedron in $\mathbb{R}^{\hat{D}}$ (one copy per row of $\mathbf{Z}_1$). At a vertex of this polyhedron, at most $\hat{k}$ coordinates per row are non-zero, determined by a $\leq \hat{k} \times \hat{k}$ system with integer coefficients from $\mathbf{C}$. By Cramer's rule, these values are rationals with numerator and denominator at most $D^{O(\hat{k})}$. Multiplying by the common denominator preserves all signs and thresholds, yielding integers in $[-L, L]$.

State/Subinstance Space: For every $d \in [0, \hat{D}]$ and matrix $\mathbf{S} \in [-L', L']^{M' \times \hat{k}}$ where $L' = D \cdot L = D^{O(\hat{k})}$, let $\text{Search}(d, \mathbf{S})$ be a boolean value indicating whether:

- $\exists \mathbf{z}_1 \dots \mathbf{z}_d \in [-L, L]^{M'}$: (Pre-threshold vectors on the 1st d)
 - $\forall k \in [\hat{k}] \mathbf{s}_k = \sum_{d'=1}^d c_{d'k} \mathbf{z}_{d'}$ (Possible Pre-sums)
 - $\forall d' \in [d] (f_3(\mathbf{z}_{d'}) - y_{d'}^*)^2 \leq \epsilon$ (Correct output on the 1st d)

Recurrence Relation: This same $\text{Search}(d, \mathbf{S})$ can be recursively defined as:

- **Base Case:** $\text{Search}(0, \mathbf{S}) = \text{true}$ iff $\mathbf{S} = \mathbf{0}$.
- $\exists \mathbf{S}' \in [-L', L']^{M' \times \hat{k}} \exists \mathbf{z}_d \in [-L, L]^{M'}$: (Pre-sums and pre-threshold vectors on the d^{th})
 - $\text{Search}(d-1, \mathbf{S}')$ (Induction hypothesis)
 - $\forall k \in [\hat{k}] \mathbf{s}_k = \mathbf{s}'_k + c_{dk} \mathbf{z}_d$ (Possible Pre-sums)
 - $(f_3(\mathbf{z}_d) - y_d^*)^2 \leq \epsilon$ (Correct output on the d^{th})

Conclusion: We accept iff:

- $\exists \mathbf{S} \in [-L', L']^{M' \times \hat{k}}$: (Pre-sums)
 - $\text{Search}(\hat{D}, \mathbf{S})$ (Induction Conclusion)
 - $\forall k \in [\hat{k}] \hat{\mathbf{z}}_{\hat{D}+k} = \text{Threshold}(\mathbf{s}_k)$ (Correct Dependencies)

Running Time: We enumerate over $2^{M'\hat{k}}$ matrices $\hat{\mathbf{Z}}_2 \in \{0, 1\}^{M' \times \hat{k}}$. For each, the number of states/sub-instances $\text{Search}(d, \mathbf{S})$ is $\hat{D} \times (2L' + 1)^{M'\hat{k}} = D^{O(M'\hat{k}^2)}$ because $L' = D^{O(\hat{k})}$. For each, we try each $\mathbf{z}_d \in [-L, L]^{M'}$, computing $\mathbf{S}' = \mathbf{S} - c_d \mathbf{z}_d$ and looking up $\text{Search}(d-1, \mathbf{S}')$. For a total time of $D^{O(M'\hat{k}^2)}$.

Output: The $\mathbf{Z}_1 \in [-L, L]^{M' \times \hat{D}}$ for the solution can be recovered by backtracking. The weight matrix $\mathbf{W}$ can be found by solving the linear program with sign constraints $\forall d \in [D], \forall m' \in [M']$: $\text{Threshold}(\mathbf{w}_m^T \mathbf{x}_d) = \hat{z}_{m', d}$. The DP guarantees that such a feasible point exists. $\square$

Part IV

Conclusions

21 Discussion

Implications for Transfer Learning. Our theoretical setup—training the weights $\mathbf{W}$ of a final layer while keeping the feature map $f_1(\mathbf{a})$ fixed—closely mirrors the modern practice of **Transfer Learning** and **Linear Probing** [Arora et al., 2019, Jacot et al., 2018].

- **The Role of Depth (f_1):** Deep pre-training maps raw data into a high-dimensional feature space where the representations $\mathbf{x}_d$ are linearly independent.
- **The Role of Width ($M \geq D$):** Once a sufficiently wide feature space is found, gradient descent is guaranteed to reach zero training error, provided f_1 is fixed and the sensitivities α_d stay bounded away from zero (Theorem 1, Remark 1). Without this width, training can become NP-complete (Part III).

Scope and Limitations. Our main theorems hold f_1 and f_3 fixed while training $\mathbf{W}$ —a simplification shared with lazy/NTK analyses. As $\mathbf{W}$ changes, the pre-activations $\mathbf{Z}_t = \mathbf{W}_t \mathbf{X}$ change, and hence $\mathbf{R}_t$ changes. We prove (Reversed-Order Rank Corollary, Section 16) that for random f_3 , the set of pre-activations $\mathbf{Z}$ where $\mathbf{R}$ loses rank is a measure-zero analytic subvariety. Thus, the gradient-descent trajectory generically maintains full rank of $\mathbf{R}_t$ at every iteration. However, the trajectory can *approach* this bad set, making $\lambda_{\min}^+$ small and convergence slow—this is consistent with the rate drops observed in our simulations (Section 12). Composing the one-step guarantee $\mathbf{e}_{t+1} = (\mathbf{I} - 2\eta \mathbf{R}_t) \mathbf{e}_t$ across many iterations with varying $\mathbf{R}_t$ remains an open problem.

Consistency with NP-Completeness. One might worry that the “no traps” results contradict the NP-completeness of training. They do not. The NP reduction (Part III) uses $f_3 = \text{AND}(\text{Threshold}(\mathbf{z}_d))$ with $\text{Rank}(\mathbf{X}) = \hat{D} = D/3$, which violates all three “no traps” conditions (non-random f_3 , $MM' < D$, $\alpha_d = 0$ at hard thresholds). Moreover, different valleys of the NP landscape correspond to different $\mathbf{W}$, which produce different $\mathbf{Z}$ and hence different $\mathbf{R}$ —each valley has its own ghost subspace and its own frozen error. The ghost error being “fixed” holds for a single $\mathbf{R}$; different local minima have different $\mathbf{R}$ matrices and can therefore have different error levels.

22 Open Problems

1. **A Floor on $\lambda_{\min}(\mathbf{R}_t)$ Along Training.** This is the central open problem. Rank preservation does not give convergence (Remark 2). For $M < D \leq MM'$ with random f_1 , $\mathbf{W}_0$, and f_3 , show that $\lambda_{\min}(\mathbf{R}_t) \geq c \alpha_{\min}$ for all t (at least for realizable targets)

whenever $MM' \geq CD$, for constants $C > 1$ and $c > 0$ that do not depend on D . In the wide case this holds (Remark 1).

2. **Time-Varying f_1 , f_3 , and $\mathbf{R}$.** Unlike Part I, Part II assumes f_1 and f_3 are fixed in order to make the rate matrix $\mathbf{R}_t$ easier to understand. Saying more when gradient descent learns these is an open problem. Some of our results like $\mathcal{E}_{dt} = \prod_{t'} (1 - 4\eta\alpha_{dt'})\mathcal{E}_{d0}$ cleanly express the effect of a changing sensitivity α_{dt} . For other results like Theorem 2.4 (Managed Noise) we do not really understand how a changing sensitivity affects things. In particular, composing one-step bounds $\mathbf{e}_{t+1} = (\mathbf{I} - 2\eta\mathbf{R}_t)\mathbf{e}_t$ across many iterations with varying eigenstructure and proving that per-point rates stay close to our predictions remains open. Our model allows f_1 and f_3 to contain transformers, but learning their parameters is beyond the current analysis.
3. **Tightening the Managed-Noise Bound.** First, derive the coupling hypothesis of Section 14 from the random-network model; it is currently heuristic. Our managed-noise bound requires $M \geq \frac{4\bar{\alpha}}{\alpha_{\min}}D$. Is the ratio $\bar{\alpha}/\alpha_{\min}$ tight, or is it an artifact of the Cauchy–Schwarz step in the proof?
4. **Minimization in the Unrealizable Regime.** Our $2^{\Omega(\hat{k})}$ lower and $D^{O(M'\hat{k}^2)}$ upper bounds on the time for $\min_{\mathbf{W}} \max_d (f_3(\mathbf{z}_d) - y_d^*)^2$ could be tightened. More importantly, the complexity of $\min_{\mathbf{W}} \sum_d (f_3(\mathbf{z}_d) - y_d^*)^2$ is completely open. Our upper bound avoids this difficulty because it requires each of the $\hat{D}$ independent data points to match their target labels y_d^* within a fixed ϵ . Any algorithm minimizing the sum must negotiate trade-offs across these points, likely forcing an enumeration over $2^{\Omega(\hat{D})}$ possible output combinations to identify the global optimum. We conjecture that global error minimization may be NP-complete even for a rank deficit of $\hat{k} = 1$.
5. **Marchenko–Pastur Law for $\mathbf{R}$.** We conjecture that the eigenvalue spread of $\mathbf{R}$ satisfies $\text{CV}(\lambda) \approx \sqrt{D/(MM')}$, mirroring the Marchenko–Pastur law for $\mathbf{X}^T\mathbf{X}$. This is supported by simulations but a proof would pin down $\lambda_{\min}^+$ and thereby the precise convergence rate.
6. **SGD and Ghost Absorption.** Do the simple models in Theorem 2.7.7 accurately capture how rate-matrix rotation and stochastic gradient descent cause the ghost error to decay to zero? Stochastic gradient descent perturbs $\mathbf{W}_t$ at each step, which can rotate $\mathbf{R}_t$ and its null space. Any resulting increase in non-ghost error decays exponentially (at rate $\geq \lambda_{\min}^+$), while ghost error absorbed into non-ghost channels is permanently reduced. Does SGD converge to zero total error in the underparameterized regime?

23 Conclusion

The convergence and landscape results of this paper reinforce each other across four regimes:

- **Wide** ($M \geq D$, random f_1): $\mathbf{X}$ has full rank (w.p.1), so $k = 0$. Suboptimal local minima exist but GD avoids them (Theorem 1): with f_1 fixed, the error decays to zero (Remark 1). While the errors are comparable, each point’s error decays at its own rate α_d (Theorem 2.3).
- **Overparameterized** ($MM' \geq D$, random f_1 and f_3): $\mathbf{R}$ has full rank (w.p.1), so $k = 0$. Suboptimal local minima exist and GD never steps onto them ($\mathbf{R}_t$ keeps full rank) (Theorem 1). Whether the error then decays to zero is open; under a coupling hypothesis it decays at a collective rate of at least $\frac{1}{2}\alpha_{\min}$ (Theorem 2.4).

- **Underparameterized** ($MM' < D$): Under a fixed $\mathbf{R}$, the non-ghost error still decays to zero; the $k = D - MM'$ ghost channels receive zero gradient under a given $\mathbf{R}$. Widths beyond $\text{Rank}(\mathbf{X})$ provide no additional benefit to the training loss.
- **NP-Complete**: At the other extreme, when α_d vanishes and the data representation has rank deficit $\hat{k}$, training becomes NP-complete—with a $2^{\Omega(\hat{k})}$ lower bound and a $D^{O(M'\hat{k}^2)}$ algorithm.

Most of these results hold for general activations (rank preservation needs analytic ones) without the massive-width or ReLU-only assumptions common in the literature, though we assume the sensitivities α_d stay bounded away from zero. The recurring lesson is that training difficulty is not a cliff but a dial: the activation sensitivity α_d , the rank deficit k of $\mathbf{R}$, and the rank deficit $\hat{k}$ of $\mathbf{X}$ are faucets that smoothly interpolate between the tractable and the intractable.

Acknowledgments

It is a bit ironic writing a paper about gradient descent when I got so much help from machines trained this way. I would also like to thank Arthur Jacot and the anonymous reviewers of my first attempt at a gradient descent paper, whose feedback pointed me to much of the related work cited here.

I figure there is a small window of time in which AI assistants and I can write papers **together**. This is my third one. Three years ago they were not helpful to me. I watch the incredible progress. Three years from now, I won't be helpful to them.

References

- Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via overparameterization. *ICML*, pages 242–252, 2019.
- G. W. Anderson, A. Guionnet, and O. Zeitouni. *An Introduction to Random Matrices*. Cambridge University Press, 2010. Section 2.5.
- S. Arora, S. S. Du, W. Hu, Z. Li, and R. Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *ICML*, 2019.
- P. Auer, M. Herbster, and M. K. Warmuth. Exponentially many local minima for single neurons. *NIPS*, 8:316–322, 1996.
- Z. Bai and J. W. Silverstein. *Spectral Analysis of Large Dimensional Random Matrices*. Springer, 2nd edition, 2010.
- A. Bloemendal, A. Knowles, H.-T. Yin, and J. Yin. On the principal components of sample covariance matrices. *Probability Theory and Related Fields*, 164(1–2):459–552, 2016. (arXiv:1404.0788, 2014).
- A. L. Blum and R. L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 5(1):117–127, 1992.

- L. Chizat and F. Bach. On the global convergence of gradient descent for overparameterized models using optimal transport. *NeurIPS*, 31, 2018.
- Lénaïc Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- Tian Ding, Dawei Li, and Ruoyu Sun. Suboptimal local minima exist for wide neural networks with smooth activations. *Mathematics of Operations Research*, 47(4):2784–2814, 2022.
- S. S. Du, X. Zhai, B. Póczos, and A. Singh. Gradient descent provably optimizes overparameterized neural networks. *ICLR*, 2019.
- C. Daniel Freeman and Joan Bruna. Topology and geometry of half-rectified network optimization. In *International Conference on Learning Representations (ICLR)*, 2017.
- V. Froese and C. Hertrich. The computational complexity of ReLU network training parameterized by data dimensionality. *JAIR*, 75, 2022.
- A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *NeurIPS*, pages 8571–8580, 2018.
- Arthur Jacot, Franck Gabriel, and Clément Hongler. The asymptotic spectrum of the Hessian of DNN throughout training. arXiv preprint arXiv:1910.02875, 2019.
- H. Jiang. Why learning of large-scale neural networks behaves like convex optimization. *ArXiv:1903.02140*, 2019.
- Z. Ji and M. Telgarsky. The implicit bias of gradient descent on nonseparable data. *COLT*, pages 1772–1798, 2019.
- Kenji Kawaguchi. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.
- S. Khalife and A. Basu. On the computational complexity of training ReLU networks. *JAIR*, 74:1001–1024, 2022.
- A. Knowles and H.-T. Yin. Eigenvector distribution of Wigner matrices. *Probability Theory and Related Fields*, 155(3–4):543–582, 2013.
- M. L. Mehta. *Random Matrices*. Academic Press, 3rd edition, 2004. Chapter 3.
- Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.
- Q. Nguyen and M. Hein. The loss surface of deep and wide neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, PMLR 70, pages 2603–2612, 2017.
- Q. Nguyen and M. Hein. Optimization landscape and expressivity of deep CNNs. *ICML*, 2018.

- Q. Nguyen. On the proof of global convergence of gradient descent for deep ReLU networks with linear widths. *ICML*, 2021.
- S. O’Rourke, V. Vu, and K. Wang. Eigenvectors of random matrices: a survey. *Journal of Combinatorial Theory, Series A*, 144:361–442, 2016.
- A. Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8:143–195, 1999.
- C. Song, A. Ramezani-Kebrya, T. Pethick, A. Eftekhari, and V. Cevher. Subquadratic overparameterization for shallow neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro. The implicit bias of gradient descent on separable data. *JMLR*, 19(70):1–57, 2018.
- Luca Venturi, Afonso S. Bandeira, and Joan Bruna. Spurious valleys in one-hidden-layer neural network optimization landscapes. *Journal of Machine Learning Research*, 20(133):1–34, 2019.
- C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2017.