Theoretical Justification of Deep Learning

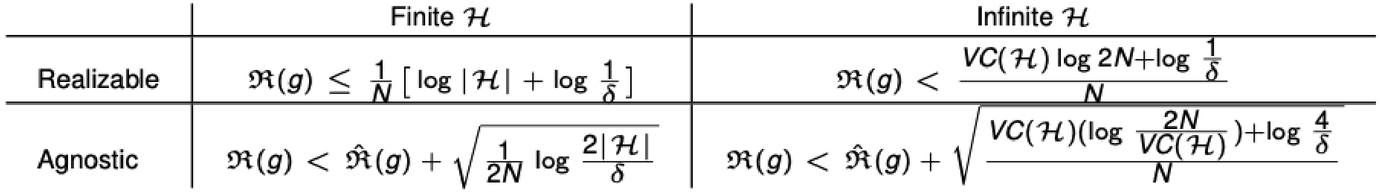
Generalization and Optimization

Oualid Missaoui

Agenda

- Standard/classic statistical learning theory (SLT)
- Double Descent and limitations of classic SLT
- Justification of Double Descent in linear models
- Linearization of Neural Networks (NTK)
- Optimization: SGD Implicit Bias
- Spectral Theory for Deep Network Generalization
- References

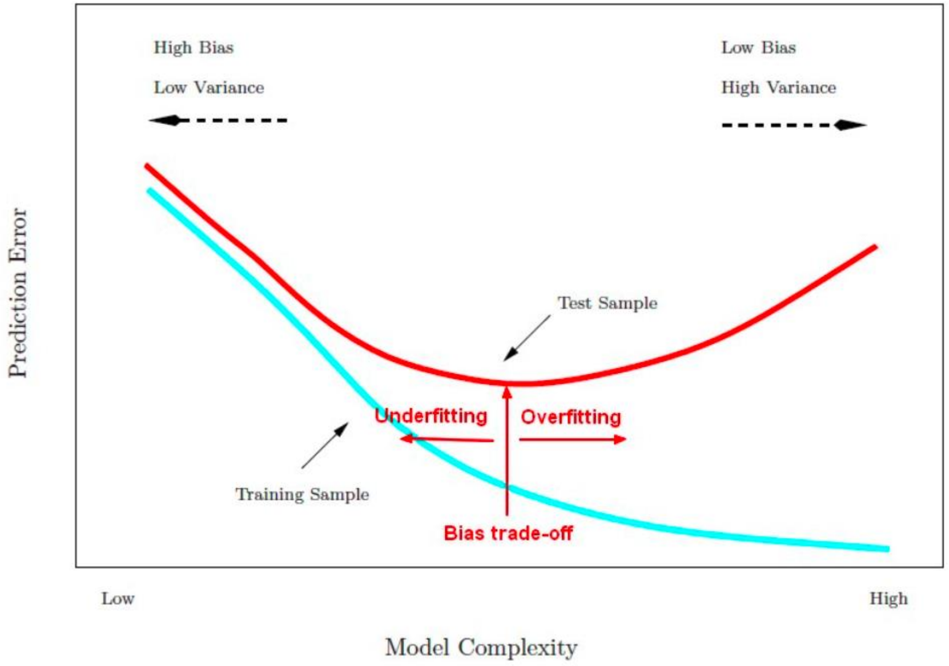
Classic/Standard Statistical Learning Theory



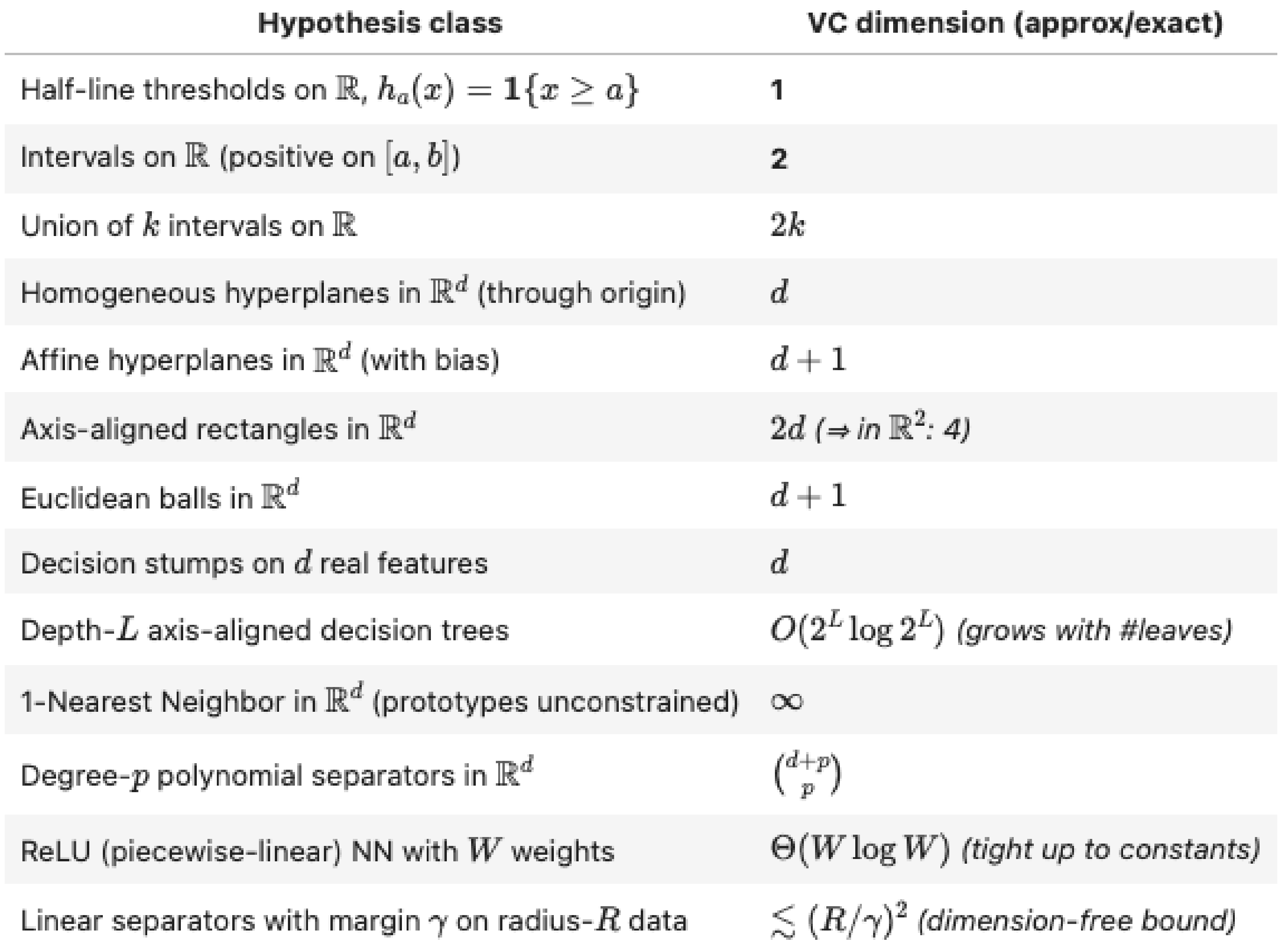
	Finite $\mathcal{H}$	Infinite $\mathcal{H}$
Realizable	$\mathfrak{R}(g) \le \frac{1}{N}\left[\log\|\mathcal{H}\| + \log\frac{1}{\delta}\right]$	$\mathfrak{R}(g) < \frac{VC(\mathcal{H})\log 2N + \log\frac{1}{\delta}}{N}$
Agnostic	$\mathfrak{R}(g) < \hat{\mathfrak{R}}(g) + \sqrt{\frac{1}{2N}\log\frac{2\|\mathcal{H}\|}{\delta}}$	$\mathfrak{R}(g) < \hat{\mathfrak{R}}(g) + \sqrt{\frac{VC(\mathcal{H})(\log\frac{2N}{VC(\mathcal{H})}) + \log\frac{4}{\delta}}{N}}$

where $VC(\mathcal{H})$ is the Vapnik-Chervonenkis dimension of the hypothesis space $\mathcal{H}$. What is $VC(\mathcal{H})$ dimension ?

- A dichotomy of a set S is a partition of S into two disjoint subsets
- A set of instances S is shattered by a hypothesis space $\mathcal{H}$ if and only if for every dichotomy of S there exists some hypothesis in $\mathcal{H}$ consistent with this dichotomy.
- The Vapnik-Chervonenkis dimension $VC(\mathcal{H})$ of hypothesis space $\mathcal{H}$ defined over instance space $\mathcal{X}$ is the size of the largest finite subset of $\mathcal{X}$ shattered by $\mathcal{H}$. If arbitrary large finite sets of $\mathcal{X}$ can be shattered by $\mathcal{H}$, then $VC(\mathcal{H}) = \infty$



VC Dimension



Hypothesis class	VC dimension (approx/exact)
Half-line thresholds on $\mathbb{R}$, $h_a(x) = \mathbf{1}\{x \geq a\}$	**1**
Intervals on $\mathbb{R}$ (positive on $[a, b]$)	**2**
Union of k intervals on $\mathbb{R}$	$2k$
Homogeneous hyperplanes in $\mathbb{R}^d$ (through origin)	d
Affine hyperplanes in $\mathbb{R}^d$ (with bias)	$d + 1$
Axis-aligned rectangles in $\mathbb{R}^d$	$2d$ *(⇒ in $\mathbb{R}^2$: 4)*
Euclidean balls in $\mathbb{R}^d$	$d + 1$
Decision stumps on d real features	d
Depth-L axis-aligned decision trees	$O(2^L \log 2^L)$ *(grows with #leaves)*
1-Nearest Neighbor in $\mathbb{R}^d$ (prototypes unconstrained)	∞
Degree-p polynomial separators in $\mathbb{R}^d$	$\binom{d+p}{p}$
ReLU (piecewise-linear) NN with W weights	$\Theta(W \log W)$ *(tight up to constants)*
Linear separators with margin γ on radius-R data	$\lesssim (R/\gamma)^2$ *(dimension-free bound)*

VC measures worst-case shattering power; higher VC implies a greater sample requirement or stronger regularization for good generalization.

Double Descent (1)

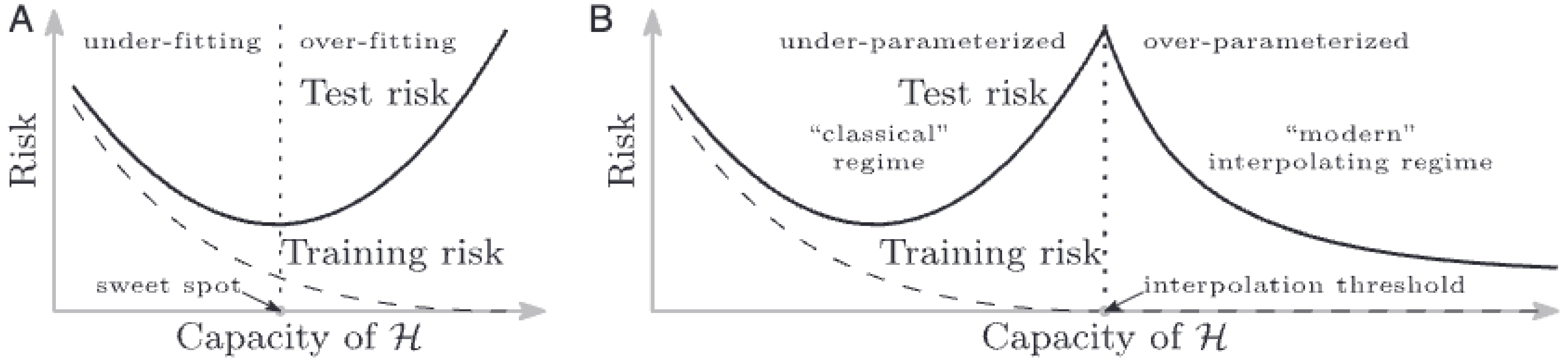


Fig. 1. Curves for training risk (dashed line) and test risk (solid line). (*A*) The classical U-shaped risk curve arising from the bias–variance trade-off. (*B*) The double-descent risk curve, which incorporates the U-shaped risk curve (i.e., the "classical" regime) together with the observed behavior from using high-capacity function classes (i.e., the "modern" interpolating regime), separated by the interpolation threshold. The predictors to the right of the interpolation threshold have zero training risk.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, Soumik Mandal, Reconciling modern machine-learning practice and the classical bias-variance trade-off

Double Descent (2)

- As model capacity increases from small to moderate, test error first decreases and then increases.
- There is a point where the model exactly fits the training data while test error is near its highest.
- Beyond this point, further increasing capacity causes test error to decrease again.
- Training error keeps decreasing with capacity and remains near zero after exact fit.
- The behavior therefore is not a single U-shape but a “double descent” in test error as capacity grows.
- Models that are far larger than needed to interpolate the data can exhibit lower test error than smaller models.
- The location of the peak and the rate of the second descent depend on the task, data, and model family.

Double Descent (3)

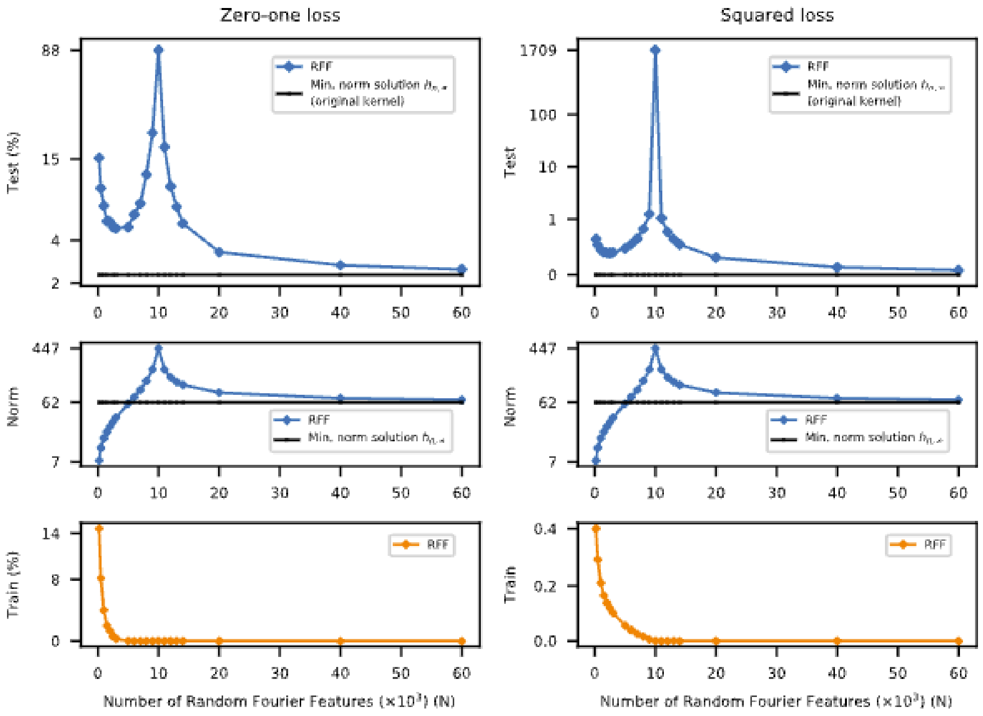


Fig. 2. Double-descent risk curve for the RFF model on MNIST. Shown are test risks (log scale), coefficient ℓ_2 norms (log scale), and training risks of the RFF model predictors $h_{n,N}$ learned on a subset of MNIST ($n = 10^4$, 10 classes). The interpolation threshold is achieved at $N = 10^4$.

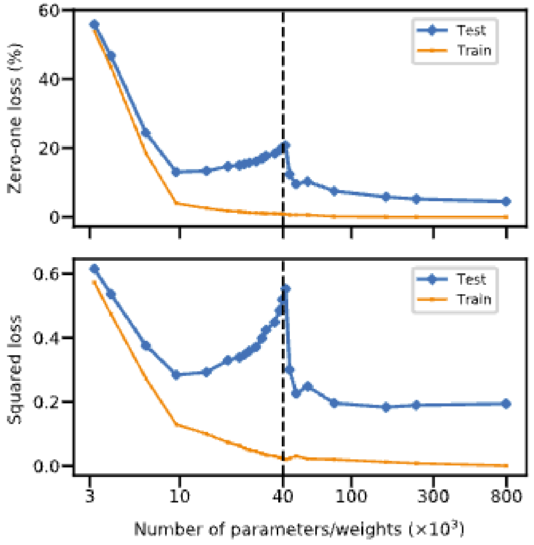


Fig. 3. Double-descent risk curve for a fully connected neural network on MNIST. Shown are training and test risks of a network with a single layer of H hidden units, learned on a subset of MNIST ($n = 4 \cdot 10^3$, $d = 784$, $K = 10$ classes). The number of parameters is $(d+1) \cdot H + (H+1) \cdot K$. The interpolation threshold (black dashed line) is observed at $n \cdot K$.

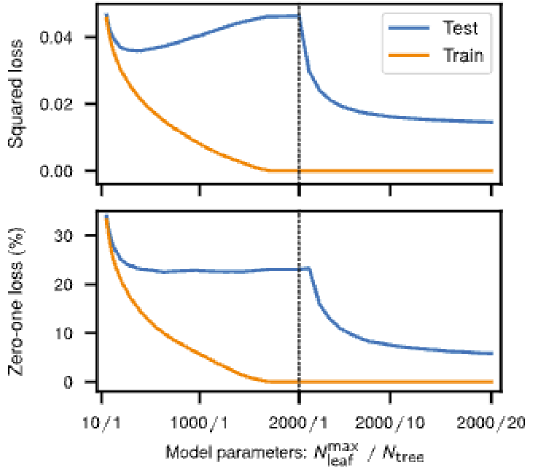


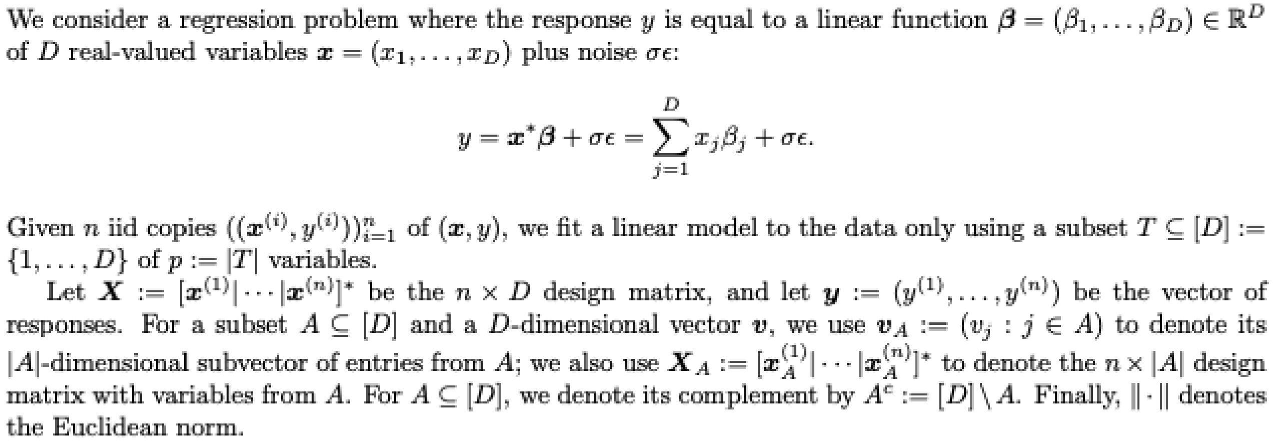
Fig. 4. Double-descent risk curve for random forests on MNIST. The double-descent risk curve is observed for random forests with increasing model complexity trained on a subset of MNIST ($n = 10^4$, 10 classes). Its complexity is controlled by the number of trees N_{tree} and the maximum number of leaves allowed for each tree $N_{\text{leaf}}^{\text{max}}$.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, Soumik Mandal, Reconciling modern machine-learning practice and the classical bias-variance trade-off

Double Descent (4)

- The **double-descent pattern** shows up in **all three**: a linear model (with random features), a **random forest**, and a **neural network**.
- For each model family, **training error** falls to (near) zero as complexity increases and stays near zero afterward.
- For each model family, **test error peaks around the interpolation threshold**—the point where the model first exactly fits the training data.
- **Linear model:** increasing the number of random features moves from under- to over-parameterized and produces the same peak-then-drop in test error.
- **Neural network:** growing hidden units/parameters shows the same peak at interpolation followed by a second descent in test error.
- **Random forest:** increasing trees/allowed depth shows the same pattern—test error rises to a peak and then declines with more capacity.
- Across losses (zero-one and squared), the qualitative behavior is the **same** for all three model classes.
- The **peak location** and **descent rate** differ by model family and the chosen complexity knob, but the **overall pattern is consistent**.

Justification of Double Descent in linear regression (1)

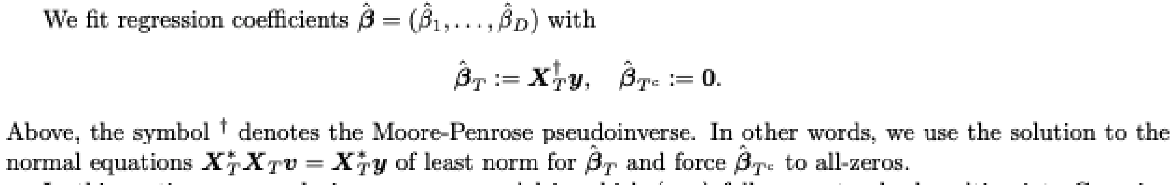


We consider a regression problem where the response y is equal to a linear function $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_D) \in \mathbb{R}^D$ of D real-valued variables $\boldsymbol{x} = (x_1, \ldots, x_D)$ plus noise $\sigma\epsilon$:

$$y = \boldsymbol{x}^* \boldsymbol{\beta} + \sigma\epsilon = \sum_{j=1}^{D} x_j \beta_j + \sigma\epsilon.$$

Given n iid copies $((\boldsymbol{x}^{(i)}, y^{(i)}))_{i=1}^{n}$ of $(\boldsymbol{x}, y)$, we fit a linear model to the data only using a subset $T \subseteq [D] := \{1, \ldots, D\}$ of $p := |T|$ variables.

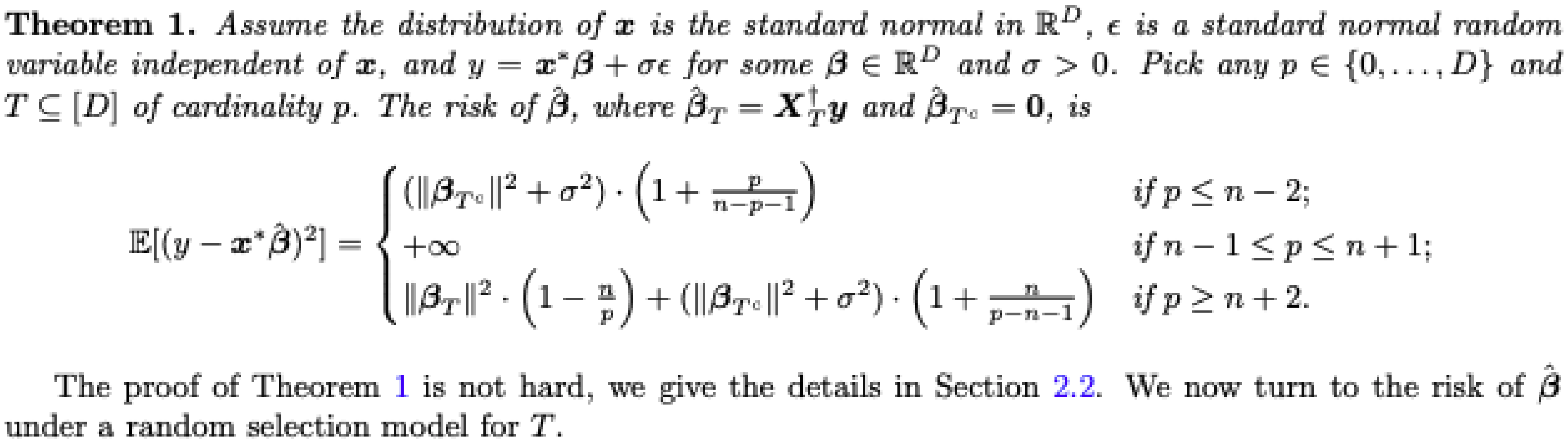
Let $\boldsymbol{X} := [\boldsymbol{x}^{(1)}|\cdots|\boldsymbol{x}^{(n)}]^*$ be the $n \times D$ design matrix, and let $\boldsymbol{y} := (y^{(1)}, \ldots, y^{(n)})$ be the vector of responses. For a subset $A \subseteq [D]$ and a D-dimensional vector $\boldsymbol{v}$, we use $\boldsymbol{v}_A := (v_j : j \in A)$ to denote its $|A|$-dimensional subvector of entries from A; we also use $\boldsymbol{X}_A := [\boldsymbol{x}_A^{(1)}|\cdots|\boldsymbol{x}_A^{(n)}]^*$ to denote the $n \times |A|$ design matrix with variables from A. For $A \subseteq [D]$, we denote its complement by $A^c := [D] \setminus A$. Finally, $\|\cdot\|$ denotes the Euclidean norm.



We fit regression coefficients $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \ldots, \hat{\beta}_D)$ with

$$\hat{\boldsymbol{\beta}}_T := \boldsymbol{X}_T^{\dagger} \boldsymbol{y}, \quad \hat{\boldsymbol{\beta}}_{T^c} := \boldsymbol{0}.$$

Above, the symbol † denotes the Moore-Penrose pseudoinverse. In other words, we use the solution to the normal equations $\boldsymbol{X}_T^* \boldsymbol{X}_T \boldsymbol{v} = \boldsymbol{X}_T^* \boldsymbol{y}$ of least norm for $\hat{\boldsymbol{\beta}}_T$ and force $\hat{\boldsymbol{\beta}}_{T^c}$ to all-zeros.



Theorem 1. *Assume the distribution of $\boldsymbol{x}$ is the standard normal in $\mathbb{R}^D$, ϵ is a standard normal random variable independent of $\boldsymbol{x}$, and $y = \boldsymbol{x}^*\boldsymbol{\beta} + \sigma\epsilon$ for some $\boldsymbol{\beta} \in \mathbb{R}^D$ and $\sigma > 0$. Pick any $p \in \{0, \ldots, D\}$ and $T \subseteq [D]$ of cardinality p. The risk of $\hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}_T = \boldsymbol{X}_T^{\dagger}\boldsymbol{y}$ and $\hat{\boldsymbol{\beta}}_{T^c} = \boldsymbol{0}$, is*

$$\mathbb{E}[(y - \boldsymbol{x}^*\hat{\boldsymbol{\beta}})^2] = \begin{cases} (\|\boldsymbol{\beta}_{T^c}\|^2 + \sigma^2) \cdot \left(1 + \frac{p}{n-p-1}\right) & \text{if } p \le n-2; \\ +\infty & \text{if } n-1 \le p \le n+1; \\ \|\boldsymbol{\beta}_T\|^2 \cdot \left(1 - \frac{n}{p}\right) + (\|\boldsymbol{\beta}_{T^c}\|^2 + \sigma^2) \cdot \left(1 + \frac{n}{p-n-1}\right) & \text{if } p \ge n+2. \end{cases}$$

The proof of Theorem 1 is not hard, we give the details in Section 2.2. We now turn to the risk of $\hat{\boldsymbol{\beta}}$ under a random selection model for T.

Mikhail Belkin, Daniel Hsu, and Ji Xu (2020) , Two models of double descent for weak features

Justification Double Descent in linear regression (2)

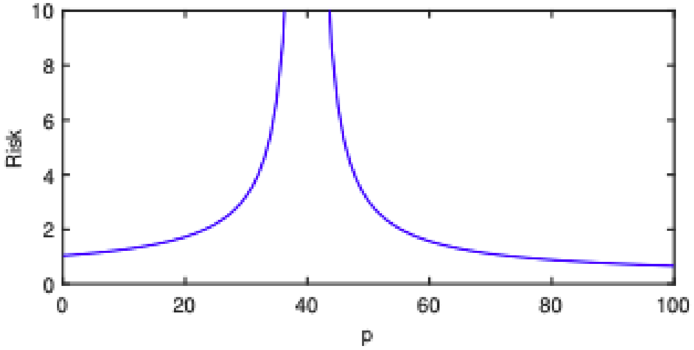


Figure 1: Plot of risk $\mathbb{E}[(y - \boldsymbol{x}^* \hat{\boldsymbol{\beta}})^2]$ as a function of p, under the random selection model of T. Here, $\|\boldsymbol{\beta}\|^2 = 1$, $\sigma^2 = 1/25$, $D = 100$, and $n = 40$.

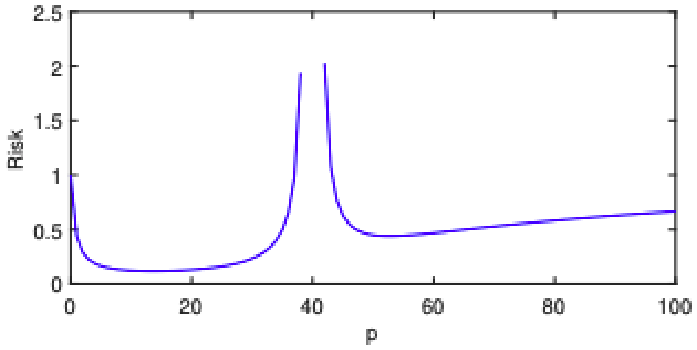


Figure 2: Plot of risk $\mathbb{E}[(y - \boldsymbol{x}^* \hat{\boldsymbol{\beta}})^2]$ as a function of p, under the “prescient” selection model of T. Here, $\|\boldsymbol{\beta}\|^2 = 1$, $\beta_j^2 \propto 1/j^2$, $\sigma^2 = 1/25$, $D = 100$, and $n = 40$.

Setup:

In this setup with n=40 and D=100, least-squares regression uses p features

Random Feature Choice:

As p increases up to n, the model overfits and risk worsens; reflecting the classical double descent without the initial descent. Interestingly, when p>n, the risk decreases again as the model benefits from overparameterization, perfectly fitting the data while becoming less volatile.

Prescient Feature Choice (strongest first)

The risk curve shows a “sweet spot” at low feature counts, resembling traditional descent, with higher risk in the interpolation regime. This happens because in a scientific feature selection setting, one cannot hand-pick informative features;’ so interpolation only helps when feature *choice isn’t biased*. **If features can be cherry-picked, good performance is achievable even with few variables.**

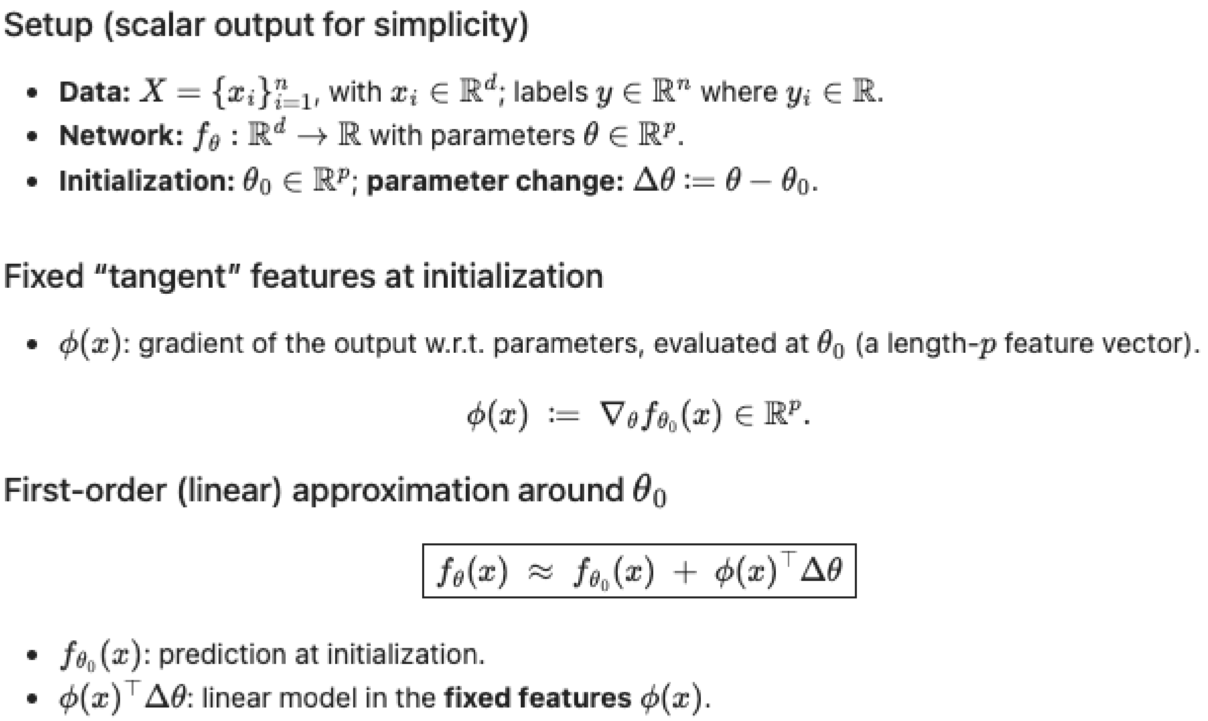
Clayton Sanford (2021) BHX19: Two models of double descent for weak features

Bridging to Neural Tangent Kernel (NTK)

- We’ve seen that linear models and kernel methods can interpolate the data; matching the double-descent pattern we observed.
- **Natural question**: Can we approximate deep nets as linear or kernel machines so we can port these insights (interpolation, minimum-norm bias, double descent) to modern networks?
- **Idea**: look at very wide networks trained with small steps; their training can be linearized around initialization and viewed through a kernel induced by the network itself.
- This leads us to the *Neural Tangent Kernel* (NTK); a framework where wide nets behave like kernel regressors with a learned, network-defined kernel.
- **Next**: define NTK, show the training dynamics, and connect it back to interpolation and generalization.

Linearization of Neural Networks

For very wide nets trained with small steps, training behaves like fitting a linear model on fixed features determined at initialization.



Setup (scalar output for simplicity)

- **Data:** $X = \{x_i\}_{i=1}^n$, with $x_i \in \mathbb{R}^d$; labels $y \in \mathbb{R}^n$ where $y_i \in \mathbb{R}$.
- **Network:** $f_\theta : \mathbb{R}^d \to \mathbb{R}$ with parameters $\theta \in \mathbb{R}^p$.
- **Initialization:** $\theta_0 \in \mathbb{R}^p$; **parameter change:** $\Delta\theta := \theta - \theta_0$.

Fixed "tangent" features at initialization

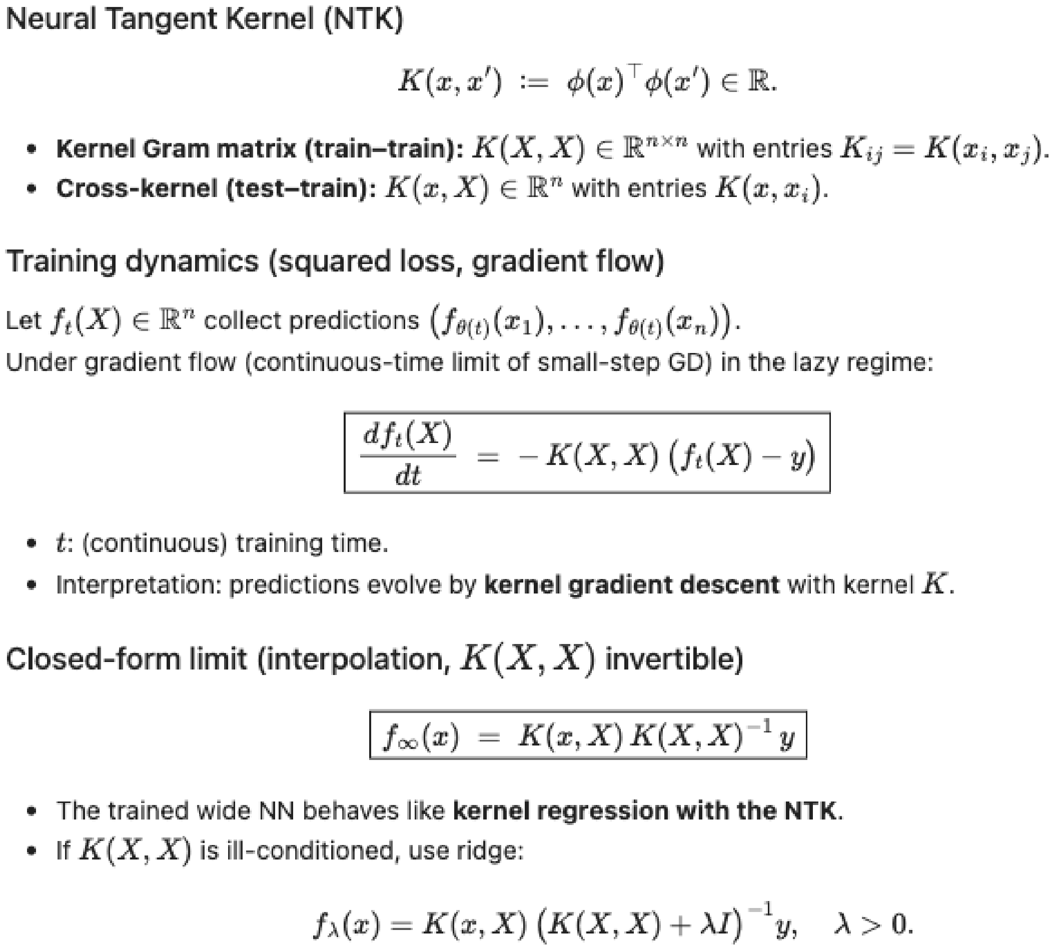
- $\phi(x)$: gradient of the output w.r.t. parameters, evaluated at θ_0 (a length-p feature vector).

$$\phi(x) \;:=\; \nabla_\theta f_{\theta_0}(x) \in \mathbb{R}^p.$$

First-order (linear) approximation around θ_0

$$\boxed{f_\theta(x) \;\approx\; f_{\theta_0}(x) \;+\; \phi(x)^\top \Delta\theta}$$

- $f_{\theta_0}(x)$: prediction at initialization.
- $\phi(x)^\top \Delta\theta$: linear model in the **fixed features** $\phi(x)$.



Neural Tangent Kernel (NTK)

$$K(x, x') \;:=\; \phi(x)^\top \phi(x') \in \mathbb{R}.$$

- **Kernel Gram matrix (train–train):** $K(X, X) \in \mathbb{R}^{n\times n}$ with entries $K_{ij} = K(x_i, x_j)$.
- **Cross-kernel (test–train):** $K(x, X) \in \mathbb{R}^n$ with entries $K(x, x_i)$.

Training dynamics (squared loss, gradient flow)

Let $f_t(X) \in \mathbb{R}^n$ collect predictions $\big(f_{\theta(t)}(x_1), \ldots, f_{\theta(t)}(x_n)\big)$.
Under gradient flow (continuous-time limit of small-step GD) in the lazy regime:

$$\boxed{\frac{d f_t(X)}{dt} \;=\; -\,K(X, X)\,\big(f_t(X) - y\big)}$$

- t: (continuous) training time.
- Interpretation: predictions evolve by **kernel gradient descent** with kernel K.

Closed-form limit (interpolation, $K(X, X)$ invertible)

$$\boxed{f_\infty(x) \;=\; K(x, X)\, K(X, X)^{-1}\, y}$$

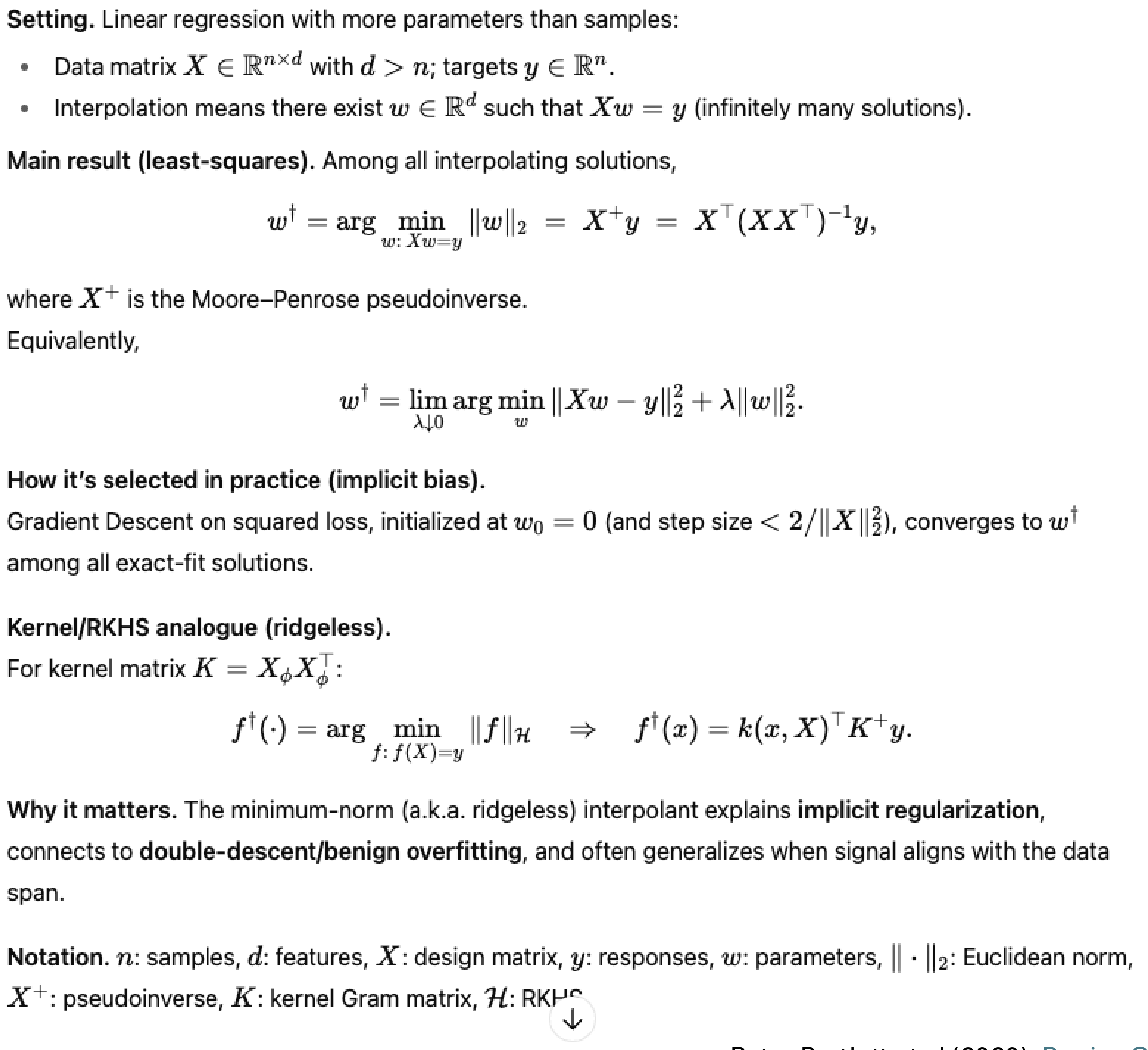
- The trained wide NN behaves like **kernel regression with the NTK**.
- If $K(X, X)$ is ill-conditioned, use ridge:

$$f_\lambda(x) = K(x, X)\,\big(K(X, X) + \lambda I\big)^{-1} y, \qquad \lambda > 0.$$

Wide + small steps ⇒ NN ≈ NTK kernel regression ⇒ double descent/benign overfitting results from linear & kernel models largely apply.

Arthur Jacot et al (2020), Neural Tangent Kernel: Convergence and Generalization in Neural Networks

Implicit Bias of Gradient Descent (1)



Setting. Linear regression with more parameters than samples:

- Data matrix $X \in \mathbb{R}^{n\times d}$ with $d > n$; targets $y \in \mathbb{R}^n$.
- Interpolation means there exist $w \in \mathbb{R}^d$ such that $Xw = y$ (infinitely many solutions).

Main result (least-squares). Among all interpolating solutions,

$$w^\dagger = \arg\min_{w:\, Xw=y} \|w\|_2 \;=\; X^+y \;=\; X^\top(XX^\top)^{-1}y,$$

where X^+ is the Moore–Penrose pseudoinverse.
Equivalently,

$$w^\dagger = \lim_{\lambda\downarrow 0} \arg\min_{w} \|Xw - y\|_2^2 + \lambda\|w\|_2^2.$$

How it's selected in practice (implicit bias).
Gradient Descent on squared loss, initialized at $w_0 = 0$ (and step size $< 2/\|X\|_2^2$), converges to $w^\dagger$ among all exact-fit solutions.

Kernel/RKHS analogue (ridgeless).
For kernel matrix $K = X_\phi X_\phi^\top$:

$$f^\dagger(\cdot) = \arg\min_{f:\, f(X)=y} \|f\|_{\mathcal{H}} \quad\Rightarrow\quad f^\dagger(x) = k(x,X)^\top K^+ y.$$

Why it matters. The minimum-norm (a.k.a. ridgeless) interpolant explains **implicit regularization**, connects to **double-descent/benign overfitting**, and often generalizes when signal aligns with the data span.

Notation. n: samples, d: features, X: design matrix, y: responses, w: parameters, $\|\cdot\|_2$: Euclidean norm, X^+: pseudoinverse, K: kernel Gram matrix, $\mathcal{H}$: RKHS

At the interpolation threshold, the least squares matrix is nearly singular. The variance term blows up, giving the peak in the risk curve.

Beyond interpolation, there are infinitely many zero training error solutions. Gradient Descent is implicitly biased to the minimum norm interpolant

the optimizer selects the minimum-norm interpolating solution, which acts as an implicit regularizer and restores generalization in the overparameterized regime.

Peter Bartlett et al (2020), Benign Overfitting in Linear Regression

Implicit Bias of Gradient Descent (2)

- On linearly separable data, plain gradient descent on logistic/cross-entropy (and any smooth, strictly decreasing loss with an exponential tail) implicitly drives the predictor toward the hard-margin SVM solution. Concretely, the weights blow up in norm but their direction converges to the L2 max-margin separator.
- Under similar assumptions, SGD also converges in direction to the L2 max-margin separator on linearly separable data with exponentially-tailed losses (e.g., logistic).
- GD/SGD pick a specific interpolating classifier: on linearly separable data with an exponentially-tailed loss (e.g., logistic), both gradient descent and SGD drive the weight norm to infinity but their direction converges to the L2 hard-margin SVM. Practically, training longer after achieving 0 training error keeps increasing the margin, which often improves test accuracy.
  - If you're doing linear (or last-layer) classification with cross-entropy, plain GD/SGD implicitly does SVM-like margin maximization—so longer training can help margins without explicit regularization.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, Nathan Srebro (2018), The implicit Bias of Gradient Descent on Separable Data

Mor Shpigel Nacson et al (2019), Stochastic Gradient Descent on Separable Data: Exact Convergence with a Fixed Learning Rate

Taxonomy of overfitting

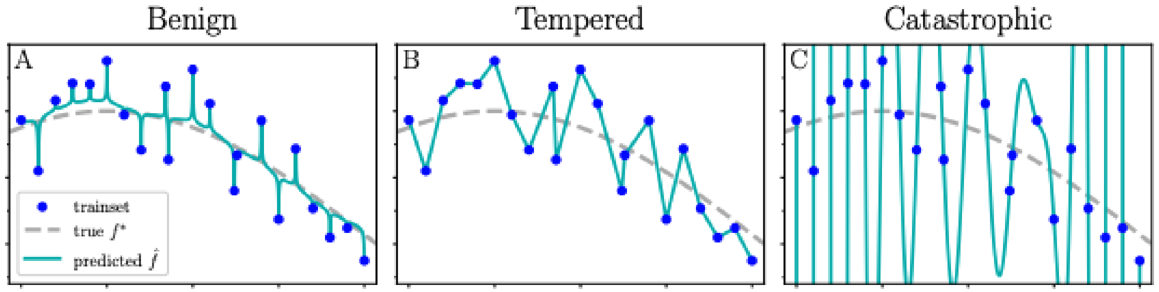
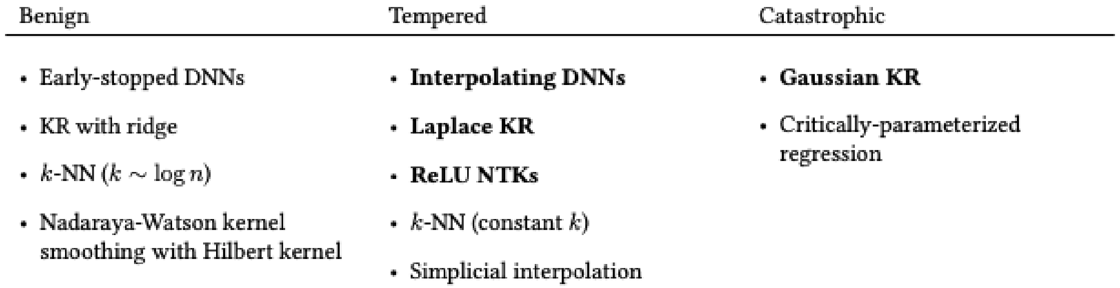
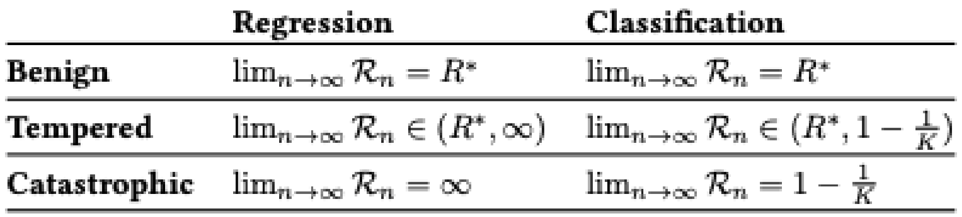


Figure 1: **As $n \to \infty$, interpolating methods can exhibit three types of overfitting. (A)** In *benign overfitting*, the predictor asymptotically approaches the ground-truth, Bayes-optimal function. Nadaraya-Watson kernel smoothing with a singular kernel, shown here, is asymptotically benign. **(B)** In *tempered overfitting*, the regime studied in this work, the predictor approaches a constant test risk greater than the Bayes-optimal risk. Piecewise-linear interpolation is asymptotically tempered. **(C)** In *catastrophic overfitting*, the predictor generalizes arbitrarily poorly. Rank-n polynomial interpolation is asymptotically catastrophic.



Benign	Tempered	Catastrophic
• Early-stopped DNNs	• **Interpolating DNNs**	• **Gaussian KR**
• KR with ridge	• **Laplace KR**	• Critically-parameterized regression
• k-NN ($k \sim \log n$)	• **ReLU NTKs**	
• Nadaraya-Watson kernel smoothing with Hilbert kernel	• k-NN (constant k)	
	• Simplicial interpolation	

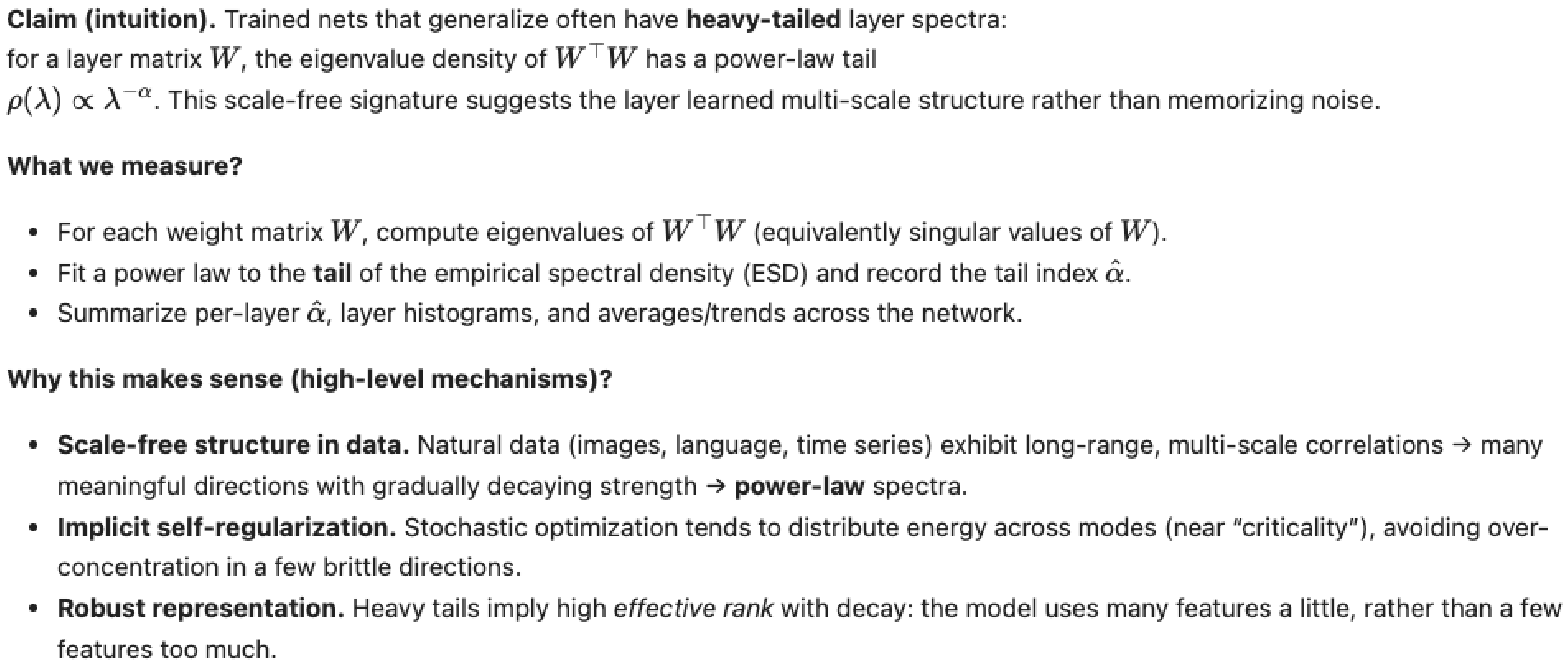
Table 2: A taxonomy of models under the three types of fitting identified in this work. **BOLD** are results from our work, others are known or folklore results.



	Regression	**Classification**
Benign	$\lim_{n\to\infty} \mathcal{R}_n = R^*$	$\lim_{n\to\infty} \mathcal{R}_n = R^*$
Tempered	$\lim_{n\to\infty} \mathcal{R}_n \in (R^*, \infty)$	$\lim_{n\to\infty} \mathcal{R}_n \in (R^*, 1 - \frac{1}{K})$
Catastrophic	$\lim_{n\to\infty} \mathcal{R}_n = \infty$	$\lim_{n\to\infty} \mathcal{R}_n = 1 - \frac{1}{K}$

Neil Mallinar et al (2024), Benign, Temered, or Catastrophic: A taxonomy of Overfitting

Why heavy-tailed weight spectra signal generalization



Claim (intuition). Trained nets that generalize often have **heavy-tailed** layer spectra:
for a layer matrix W, the eigenvalue density of $W^\top W$ has a power-law tail
$\rho(\lambda) \propto \lambda^{-\alpha}$. This scale-free signature suggests the layer learned multi-scale structure rather than memorizing noise.

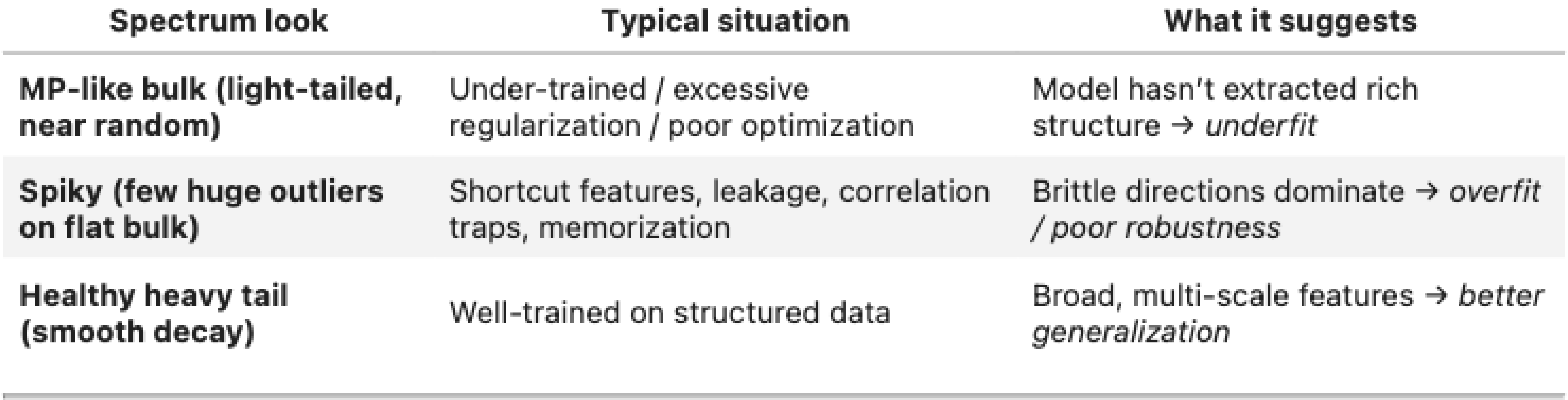
What we measure?

- For each weight matrix W, compute eigenvalues of $W^\top W$ (equivalently singular values of W).
- Fit a power law to the **tail** of the empirical spectral density (ESD) and record the tail index $\hat{\alpha}$.
- Summarize per-layer $\hat{\alpha}$, layer histograms, and averages/trends across the network.

Why this makes sense (high-level mechanisms)?

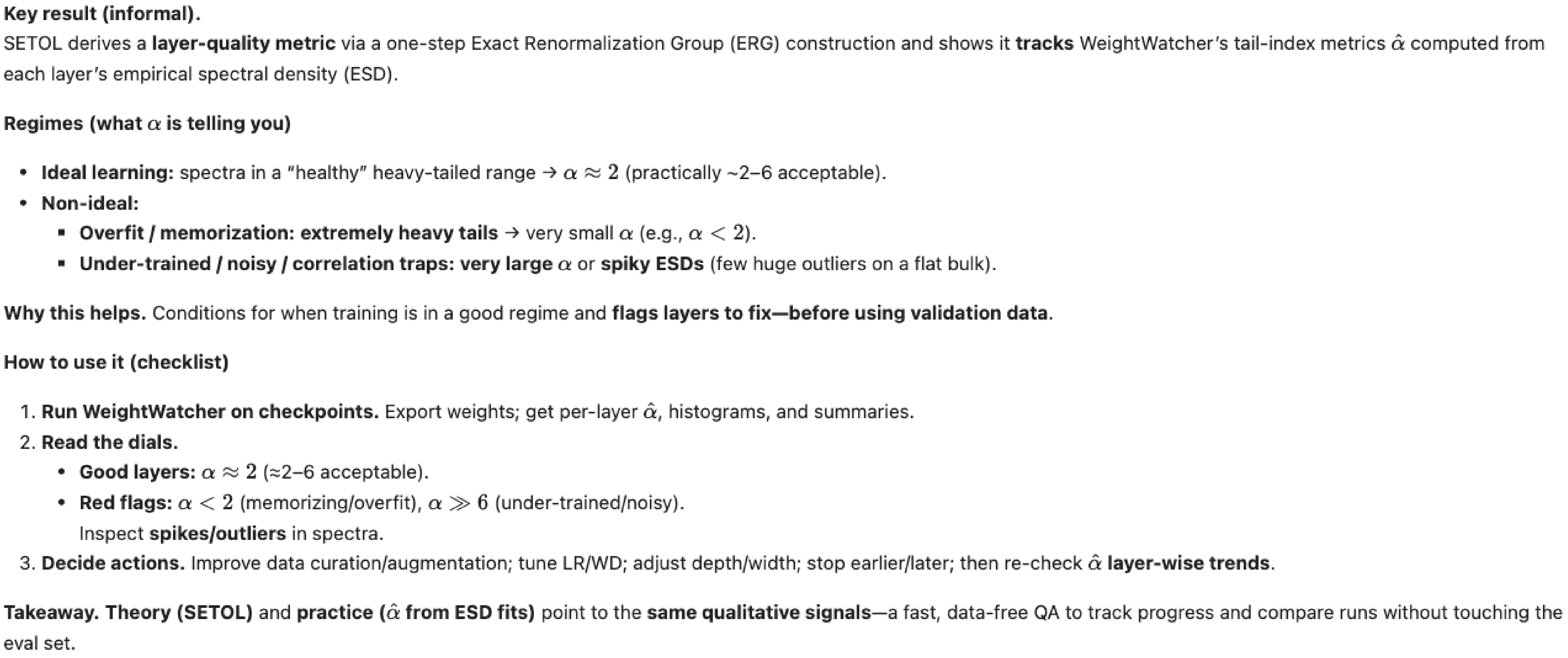
- **Scale-free structure in data.** Natural data (images, language, time series) exhibit long-range, multi-scale correlations → many meaningful directions with gradually decaying strength → **power-law** spectra.
- **Implicit self-regularization.** Stochastic optimization tends to distribute energy across modes (near "criticality"), avoiding over-concentration in a few brittle directions.
- **Robust representation.** Heavy tails imply high *effective rank* with decay: the model uses many features a little, rather than a few features too much.

If the spectrum is not heavy-tailed ?



Spectrum look	Typical situation	What it suggests
MP-like bulk (light-tailed, near random)	Under-trained / excessive regularization / poor optimization	Model hasn't extracted rich structure → *underfit*
Spiky (few huge outliers on flat bulk)	Shortcut features, leakage, correlation traps, memorization	Brittle directions dominate → *overfit / poor robustness*
Healthy heavy tail (smooth decay)	Well-trained on structured data	Broad, multi-scale features → *better generalization*

SETOL + WeightWatcher: What it says & how to use it



Key result (informal).
SETOL derives a **layer-quality metric** via a one-step Exact Renormalization Group (ERG) construction and shows it **tracks** WeightWatcher's tail-index metrics $\hat{\alpha}$ computed from each layer's empirical spectral density (ESD).

Regimes (what α is telling you)

- **Ideal learning:** spectra in a "healthy" heavy-tailed range → $\alpha \approx 2$ (practically ~2–6 acceptable).
- **Non-ideal:**
  - **Overfit / memorization: extremely heavy tails** → very small α (e.g., $\alpha < 2$).
  - **Under-trained / noisy / correlation traps: very large** α or **spiky ESDs** (few huge outliers on a flat bulk).

Why this helps. Conditions for when training is in a good regime and **flags layers to fix—before using validation data**.

How to use it (checklist)

1. **Run WeightWatcher on checkpoints.** Export weights; get per-layer $\hat{\alpha}$, histograms, and summaries.
2. **Read the dials.**
   - **Good layers:** $\alpha \approx 2$ (≈2–6 acceptable).
   - **Red flags:** $\alpha < 2$ (memorizing/overfit), $\alpha \gg 6$ (under-trained/noisy).
     Inspect **spikes/outliers** in spectra.
3. **Decide actions.** Improve data curation/augmentation; tune LR/WD; adjust depth/width; stop earlier/later; then re-check $\hat{\alpha}$ **layer-wise trends**.

Takeaway. Theory (SETOL) and **practice ($\hat{\alpha}$ from ESD fits)** point to the **same qualitative signals**—a fast, data-free QA to track progress and compare runs without touching the eval set.

Charles Martin et al, SETOL: A semi-empirical theory of (deep) learning
Spectral Diagnositcs python tool: weightwatcher

References

- Mikhail Belkin, Daniel Hsu, Siyuan Ma, Soumik Mandal, Reconciling modern machine-learning practice and the classical bias-variance trade-off
- Arthur Jacot et al (2020), Neural Tangent Kernel: Convergence and Generalization in Neural Networks
- Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, Nathan Srebro (2018), The implicit Bias of Gradient Descent on Separable Data
- Mor Shpigel Nacson et al (2019), Stochastic Gradient Descent on Separable Data: Exact Convergence with a Fixed Learning Rate
- Neil Mallinar et al (2024), Benign, Temered, or Catastrophic: A taxonomy of Overfitting
- Charles Martin and Michael Mahoney (2021), Implicit self-regularization in deep neural networks
- Charles Martin, Christopher Hinrichs (2023), SETOL: A semi-empirical theory of (deep) learning
- Oualid Missaoui (2025), Inductive Triplet Fine Tuning for Small Language Models