Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 9: Generalization in deep learning

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-556 (Fall 2022)

License Information for Mathematics of Data Slides

Outline

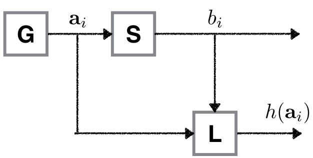
This lecture :

- The classical trade-off between model complexity and risk
- Generalization bounds via uniform convergence
- The generalization mystery in deep learning
- Implicit regularization of optimization algorithms
- Double descent curves: Generalization bounds via bias-variance decomposition
- Generalization bounds based on algorithmic stability
- Boosting

Next lecture :

- Optimization in Deep Learning

Understanding the trade-off between model complexity and expected risk



Models

Let $[\mathcal{X}_i : i = 1, \ldots]$ be a nested sequence of parameter domain, i.e., $\mathcal{X}_i \subseteq \mathcal{X}_{i+1}$. For example, let $\mathcal{X}_i$ = neural networks with i neurons.

1. $R_n(\mathbf{x}_i^\star) = \min_{\mathbf{x}\in\mathcal{X}_i} R_n(\mathbf{x})$: ERM solution over $\mathcal{X}_i$
2. $R(\mathbf{x}_i^\star)$: True risk of the ERM solution over $\mathcal{X}_i$
3. $\sup_{\mathbf{x}\in\mathcal{X}_i} |R(\mathbf{x}) - R_n(\mathbf{x})|$: Worst-case Generalization error of $\mathcal{X}_i$

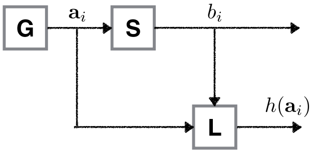
Practical performance of the ERM estimator

$$R(\mathbf{x}_i^\star) \leq \min_{\mathbf{x}\in\mathcal{X}_i} R_n(\mathbf{x}) + \sup_{\mathbf{x}\in\mathcal{X}_i} |R(\mathbf{x}) - R_n(\mathbf{x})| \tag{1}$$

As we increase the index $i \to i+1$ of the parameter domain, i.e., we choose a larger (more complex) model

1. The minimum empirical risk decreases: $\min_{\mathbf{x}\in\mathcal{X}_i} R_n(\mathbf{x}) \geq \min_{\mathbf{x}\in\mathcal{X}_{i+1}} R_n(\mathbf{x})$.
2. The generalization error increases: $\sup_{\mathbf{x}\in\mathcal{X}_i} |R(\mathbf{x}) - R_n(\mathbf{x})| \leq \sup_{\mathbf{x}\in\mathcal{X}_{i+1}} |R(\mathbf{x}) - R_n(\mathbf{x})|$.
3. What happens with the true risk $R(\mathbf{x}_i^\star)$?

Peeling the onion



Models

Let $d(\cdot,\cdot) : \mathcal{H}^\circ \times \mathcal{H}^\circ \to \mathbb{R}^+$ be a metric in an extended function space $\mathcal{H}^\circ$ that includes $\mathcal{H}$; i.e., $\mathcal{H} \subseteq \mathcal{H}^\circ$. Let

1. $h^\circ \in \mathcal{H}^\circ$ be the true, expected risk minimizing model
2. $h^\natural \in \mathcal{H}$ be the solution under the assumed function class $\mathcal{H} \subseteq \mathcal{H}^\circ$
3. $h^\star \in \mathcal{H}$ be the estimator solution
4. $h^t \in \mathcal{H}$ be the numerical approximation of the algorithm at time t

Practical performance

$$\underbrace{d(h^t, h^\circ)}_{\bar{\varepsilon}(t,n)} \leq \underbrace{d(h^t, h^\star)}_{\text{optimization error}} + \underbrace{d(h^\star, h^\natural)}_{\text{statistical error}} + \underbrace{d(h^\natural, h^\circ)}_{\text{model error}},$$

where $\bar{\varepsilon}(t,n)$ denotes the total error of the Learning Machine. We can try to

1. reduce the optimization error with computation
2. reduce the statistical error with more data samples, with better estimators, and with prior information
3. reduce the model error with flexible or universal representations

The classical trade-off between model complexity and risk

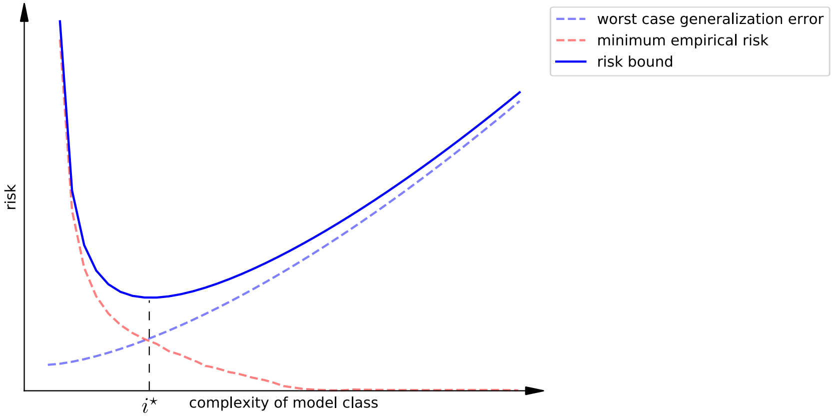


Figure: Bias-variance trade-off [15].

Occam's Razor: Simple is better than complex.

The dangers of complex function classes: sévère (cevher) overfitting

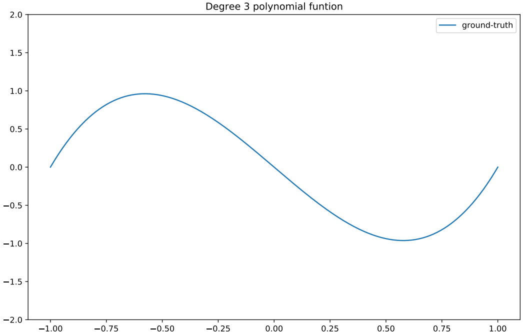
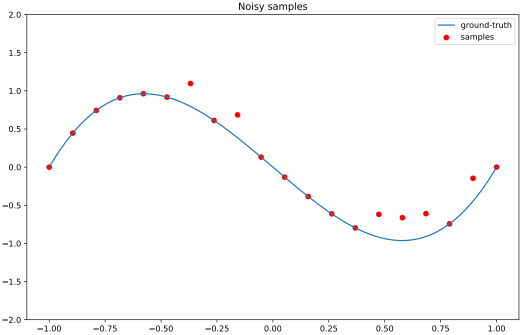


Figure: Training over a complex function class can lead to overfitting.

The dangers of complex function classes: sévère (cevher) overfitting

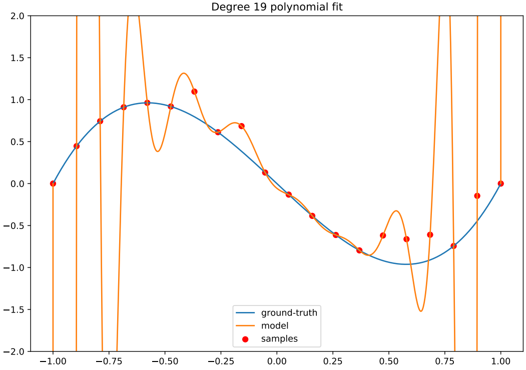


$\min_{\mathbf{x}\in\mathcal{X}} R_n(\mathbf{x}) \quad \nearrow$

$\sup_{\mathbf{x}\in\mathcal{X}} |R(\mathbf{x}) - R_n(\mathbf{x})| \quad \searrow$

Figure: Training over a complex function class can lead to overfitting.

The dangers of complex function classes: sévère (cevher) overfitting

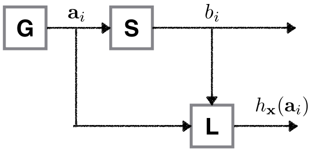


$\min_{\mathbf{x}\in\mathcal{X}} R_n(\mathbf{x}) \searrow$

$\sup_{\mathbf{x}\in\mathcal{X}} |R(\mathbf{x}) - R_n(\mathbf{x})| \nearrow$

Figure: Training over a complex function class can lead to overfitting.

Estimation of parameters vs estimation of risk



Recall the general setting

Let $R(h_{\mathbf{x}}) = \mathbb{E}L(h_{\mathbf{x}}(\mathbf{a}), b)$ be the risk function and $R_n(h_{\mathbf{x}}) = \frac{1}{n}\sum_{i=1}^{n} L(h_{\mathbf{x}}(\mathbf{a}_i), b_i)$ be the empirical estimate. Let $\mathcal{X} \subseteq \mathcal{X}^{\circ}$ be parameter domains, where $\mathcal{X}$ is known. Define

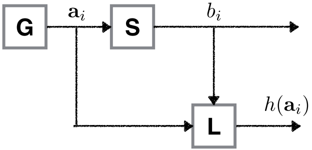
1. $\mathbf{x}^{\circ} \in \arg\min_{\mathbf{x}\in\mathcal{X}^{\circ}} R(h_{\mathbf{x}})$: true minimum risk model
2. $\mathbf{x}^{\natural} \in \arg\min_{\mathbf{x}\in\mathcal{X}} R(h_{\mathbf{x}})$: assumed minimum risk model
3. $\mathbf{x}^{\star} \in \arg\min_{\mathbf{x}\in\mathcal{X}} R_n(h_{\mathbf{x}})$: ERM solution
4. $\mathbf{x}^{t}$: numerical approximation of $\mathbf{x}^{\star}$ at time t

Nomenclature

$R_n(\cdot)$	training error
$R(\cdot)$	test error
$R(\mathbf{x}^{\natural}) - R(\mathbf{x}^{\circ})$	modeling error
$R(\mathbf{x}^{\star}) - R(\mathbf{x}^{\natural})$	excess risk
$\sup_{\mathbf{x}\in\mathcal{X}} \lvert R(\mathbf{x}) - R_n(\mathbf{x})\rvert$	generalization error
$R_n(\mathbf{x}^{t}) - R_n(\mathbf{x}^{\star})$	optimization error

	$\mathcal{X} \to \mathcal{X}^{\circ}$	$n \uparrow$	$p \uparrow$
Training error	$\searrow$	$\nearrow$	$\searrow$
Excess risk	$\nearrow$	$\searrow$	$\nearrow$
Generalization error	$\nearrow$	$\searrow$	$\nearrow$
Modeling error	$\searrow$	$=$	$\leftrightsquigarrow$
Time	$\nearrow$	$\nearrow$	$\nearrow$

What theoretical challenges in Deep Learning will we study?



Models

Let $\mathcal{X} \subseteq \mathcal{X}^\circ$ be parameter domains, where $\mathcal{X}$ is known. Define

1. $\mathbf{x}^\circ \in \arg\min_{\mathbf{x}\in\mathcal{X}^\circ} R(h_\mathbf{x})$: true minimum risk model
2. $\mathbf{x}^\natural \in \arg\min_{\mathbf{x}\in\mathcal{X}} R(h_\mathbf{x})$: assumed minimum risk model
3. $\mathbf{x}^\star \in \arg\min_{\mathbf{x}\in\mathcal{X}} R_n(h_\mathbf{x})$: ERM solution
4. $\mathbf{x}^t$: numerical approximation of $\mathbf{x}^\star$ at time t

Practical performance in Deep Learning

$$\underbrace{R(\mathbf{x}^t) - R(\mathbf{x}^\circ)}_{\bar{\varepsilon}(t,n)} \leq \underbrace{R_n(\mathbf{x}^t) - R_n(\mathbf{x}^\star)}_{\text{optimization error}} + \underbrace{2\sup_{\mathbf{x}\in\mathcal{X}} |R(\mathbf{x}) - R_n(\mathbf{x})|}_{\text{generalization error}} + \underbrace{R(\mathbf{x}^\natural) - R(\mathbf{x}^\circ)}_{\text{model error}}$$

where $\bar{\varepsilon}(t,n)$ denotes the total error of the Learning Machine. In Deep Learning applications

1. Optimization error is almost zero, in spite of non-convexity. $\Rightarrow$ lecture 10
2. Generalization error is usually small, but theory is lacking. $\Rightarrow$ lecture 9 (this one)
3. Large architectures + inductive bias might lead to small model error.

Generalization error bounds and Rademacher Complexity

Goal: Obtain generalization bounds for multi-layer, fully-connected neural networks

- We want to find high-probability upper bounds for the quantity

$$\sup_{\mathbf{x}\in\mathcal{X}} |R(\mathbf{x}) - R_n(\mathbf{x})|$$

- Need a notion of *complexity* to derive generalization bounds for infinite classes of functions

Definition (Rademacher Complexity [9])

Let $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_n\} \subseteq \mathbb{R}^p$ and let $\{v_i : i = 1, \ldots, n\}$ be independent Rademacher random variables i.e., taking values uniformly in $\{-1, +1\}$ (coin flip). Let $\mathcal{H}$ be a class of functions of the form $h : \mathbb{R}^p \to \mathbb{R}$. The Rademacher complexity of $\mathcal{H}$ with respect to A is defined as:

$$\mathcal{R}_A(\mathcal{H}) \coloneqq \mathbb{E}_v \sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} v_i h(\mathbf{a}_i).$$

Remarks:
- $\mathcal{R}_A(\mathcal{H})$ measures how well we fit random (± 1) with the output of an element of $\mathcal{H}$ on the set A.
- The derivation of Rademacher Complexity for specific function classes are in the appendix.

Fundamental theorem about the Rademacher Complexity

Theorem (See Theorem 3.3 and 5.8 in [28])

Suppose that the loss function has the form $L(h_{\mathbf{x}}(\mathbf{a}), b) = \phi(b \cdot h_{\mathbf{x}}(\mathbf{a}))$ *for a* 1*-Lipschitz function* $\phi : \mathbb{R} \to \mathbb{R}$*.*

Let $\mathcal{H}_{\mathcal{X}} := \{h_{\mathbf{x}} : \mathbf{x} \in \mathcal{X}\}$ *be a class of parametric functions* $h_{\mathbf{x}} : \mathbb{R}^p \to \mathbb{R}$*. For any* $\delta > 0$*, with probability at least* $1 - \delta$ *over the draw of an i.i.d. sample* $\{(\mathbf{a}_i, b_i)\}_{i=1}^n$*, letting* $A = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$*, the following holds:*

$$\sup_{\mathbf{x} \in \mathcal{X}} |R_n(\mathbf{x}) - R(\mathbf{x})| \leq 2\mathbb{E}_A \mathcal{R}_A(\mathcal{H}_{\mathcal{X}}) + \sqrt{\frac{\ln(2/\delta)}{2n}},$$

$$\sup_{\mathbf{x} \in \mathcal{X}} |R_n(\mathbf{x}) - R(\mathbf{x})| \leq 2\mathcal{R}_A(\mathcal{H}_{\mathcal{X}}) + 3\sqrt{\frac{\ln(4/\delta)}{2n}}.$$

Assumption is true for common losses

- $L(h_{\mathbf{x}}(\mathbf{a}), b) = \log(1 + \exp(-b \cdot h_{\mathbf{x}}(\mathbf{a}))) \Rightarrow \phi(z) := \log(1 + \exp(z))$ (logistic loss)
- $L(h_{\mathbf{x}}(\mathbf{a}), b) = \max(0, 1 - b \cdot h_{\mathbf{x}}(\mathbf{a})) \Rightarrow \phi(z) := \max(0, 1 - z)$ (hinge loss)

The complexity vs risk trade-off in practice (I)

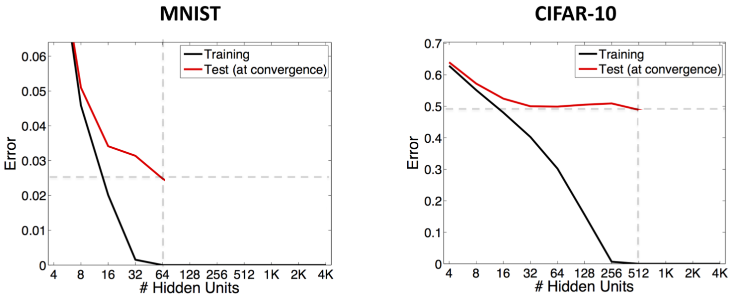


Figure: Training (empirical) and test (true) error for one-hidden-layer networks of increasing width, trained with SGD [33].

Empirical error becomes zero for a wide enough network. What should happen for even wider networks?

The complexity vs risk trade-off in practice (II)

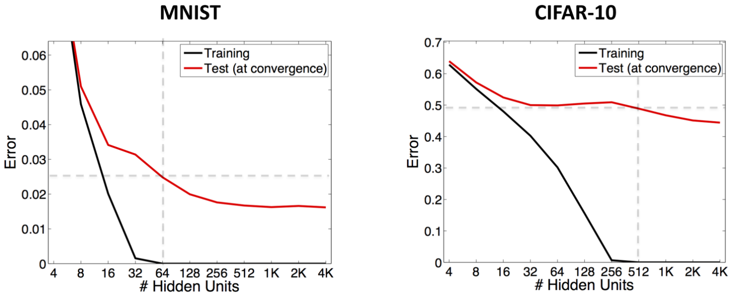


Figure: Training (empirical) and test (true) error for one-hidden-layer networks of increasing width, trained with SGD [33].

Test error continues to go down even if we keep incresing the complexity of the model!

How well do complexity measures correlate with generalization?

name	definition	correlation[1]
Frobenius distance to initialization [30]	$\sum_{i=1}^{d} \|\mathbf{X}_i - \mathbf{X}_i^0\|_F^2$	-0.263
Spectral complexity[2] [7]	$\prod_{i=1}^{d} \|\mathbf{X}_i\| \left(\sum_{i=1}^{d} \frac{\|\mathbf{X}_i\|_{2,1}^{3/2}}{\|\mathbf{X}_i\|^{3/2}} \right)^{2/3}$	$\mathbf{-0.537}$
Parameter Frobenius norm	$\sum_{i=1}^{d} \|\mathbf{X}_i\|_F^2$	0.073
Fisher-Rao [24]	$\frac{(d+1)^2}{n} \sum_{i=1}^{n} \langle \mathbf{x}, \nabla_{\mathbf{x}} \ell(h_{\mathbf{x}}(\mathbf{a}_i), b_i) \rangle$	0.078
Path-norm [34]	$\sum_{(i_0,\dots,i_d)} \prod_{j=1}^{d} \left(\mathbf{X}_{i_j, i_{j-1}} \right)^2$	$\mathbf{0.373}$

Table: Complexity measures compared in the empirical study [22], and their correlation with generalization

Complexity measures are still far from explaining generalization in Deep Learning!

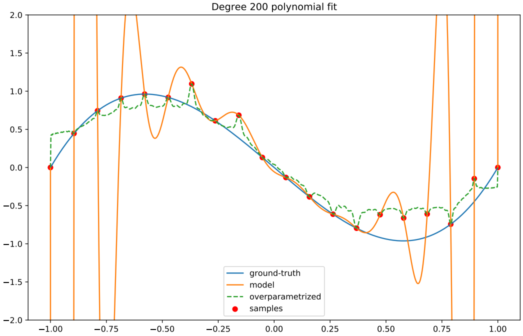
A more recent evaluation of many complexity measures is available [14].

[1] Kendall's rank correlation coefficient

[2] The definition in [22] differs slightly

The benefits of overparametrization

Overparameterization: #model parameters $\gg$ #training data



$\min_{\mathbf{x}\in\mathcal{X}} R_n(\mathbf{x}) \searrow$

$\sup_{\mathbf{x}\in\mathcal{X}} |R(\mathbf{x}) - R_n(\mathbf{x})| \searrow$

Figure: Overparametrization leads to benign overfitting.

The generalization mystery in deep learning

UNDERSTANDING DEEP LEARNING REQUIRES RE-THINKING GENERALIZATION

Chiyuan Zhang*
Massachusetts Institute of Technology
chiyuan@mit.edu

Samy Bengio
Google Brain
bengio@google.com

Moritz Hardt
Google Brain
mrtz@google.com

Benjamin Recht†
University of California, Berkeley
brecht@berkeley.edu

Oriol Vinyals
Google DeepMind
vinyals@google.com

ABSTRACT

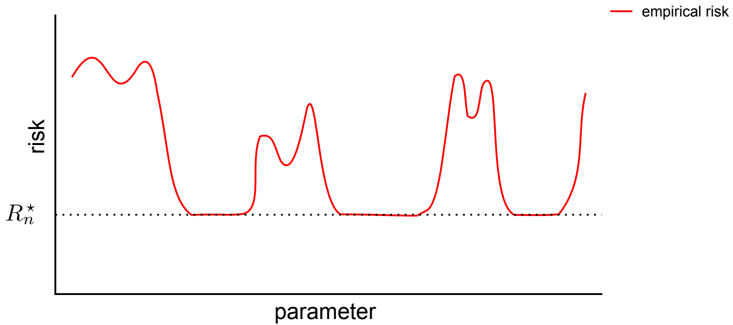
Despite their massive size, successful deep artificial neural networks can exhibit a remarkably small difference between training and test performance. Conventional wisdom attributes small generalization error either to properties of the model family, or to the regularization techniques used during training.

Through extensive systematic experiments, we show how these traditional approaches fail to explain why large neural networks generalize well in practice. Specifically, our experiments establish that state-of-the-art convolutional networks for image classification trained with stochastic gradient methods easily fit a random labeling of the training data. This phenomenon is qualitatively unaffected by explicit regularization, and occurs even if we replace the true images by completely unstructured random noise. We corroborate these experimental findings with a theoretical construction showing that simple depth two neural networks al-

A gap between theory and practice

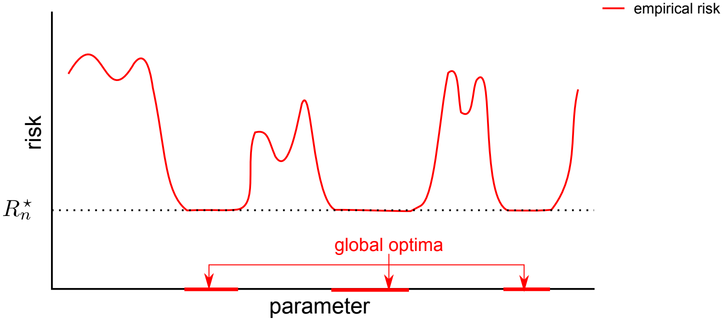
- In practice, simple algorithms like SGD can train neural networks to zero error *and* achieve low test error.
- This happens even for large and complex neural network architectures.
- Complexity measures like the Rademacher complexity suggest the opposite behaviour (overfitting)

Multiple global minimizers of the empirical risk



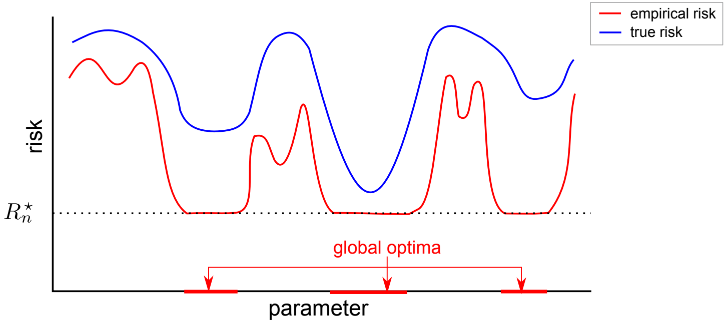
- The global minimum is $R_n^\star$

Multiple global minimizers of the empirical risk



◦ The global minimum is $R_n^\star$, but many parameters can attain such value.

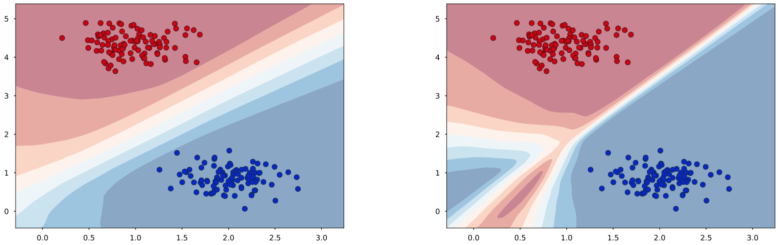
Multiple global minimizers of the empirical risk



- The global minimum is $R_n^\star$, but many parameters can attain such value.
- Each minimizer of the empirical risk might have a different true risk.

Not all global minimizers are the same

- Consider a simple $2D$ classification task, and train a neural network with fixed step-size SGD.
- The plots below correspond to two different global minimizers:



SGD never lands on the global minimum on the right! Why?

Understanding the implicit bias of optimization algorithms

- SGD seems to be *biased* towards *good* global minimizers (low true risk).
- Some optimization algorithms have an implicit bias towards certain kinds of global minimizers.
- Can we characterize this implicit bias?

Understanding the implicit bias of optimization algorithms

- SGD seems to be *biased* towards *good* global minimizers (low true risk).
- Some optimization algorithms have an implicit bias towards certain kinds of global minimizers.
- Can we characterize this implicit bias?

Definition (Algorithm)

We will refer to a function (deterministic or randomized) $\mathscr{A} : \mathcal{Z} \to \mathcal{X}$, mapping $Z \mapsto \mathscr{A}_Z$ as an *algorithm* with *input* $Z \in \mathcal{Z}$ and *output* $\mathscr{A}_Z \in \mathcal{X}$.

Example: Gradient Descent Algorithm

We denote $\text{GD}_{(T,\alpha,\mathbf{x}^0,\nabla f)} := T$-steps of GD with stepsize α, starting from $\mathbf{x}^0$, using gradient ∇f.

What is implicit regularization?

Definition (Implicit Regularization of a Deterministic Algorithm)

Consider a minimization problem

$$F^\star = \min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$$

and let $\mathscr{A}$ be a deterministic algorithm with input $Z \in \mathcal{Z}$ and output $\mathscr{A}_Z \in \mathcal{X}$.

We say that $\mathscr{A}$ solves problem (2) and has *implicit regularization* $H : \mathcal{X} \times \mathcal{Z} \to \mathbb{R}$ if

$$\mathscr{A}_Z \in \arg\min_{F(\mathbf{x})=F^\star} H(\mathbf{x}, Z).$$

Given the input $Z \in \mathcal{Z}$, the algorithm outputs a global minimizer of F that, **additionaly**, minimizes $H(\cdot, Z)$.

Implicit bias of gradient descent for linear regression

- Consider for example an underdetermined linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad \text{with } \mathbf{A} \in \mathbb{R}^{n\times p}, \quad n < p$$

- If a solution exists (i.e., $\mathbf{b} \in \mathbf{colspan}(\mathbf{A})$), then there is an *infinite number of solutions* to this system.

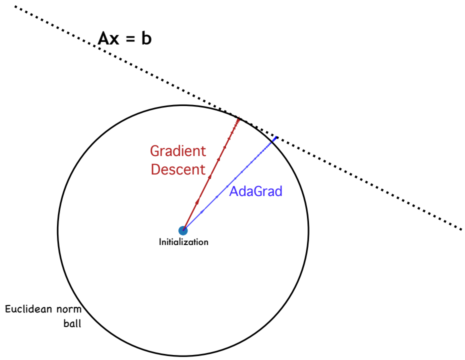
Finding a solution

To find a valid $\mathbf{x}$, we could apply one of the optimization algorithms seen in class to the convex problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$$

Among all the possible solutions, which one will the algorithm converge to ?

Same problem and same initialization vs different algorithms and different solutions

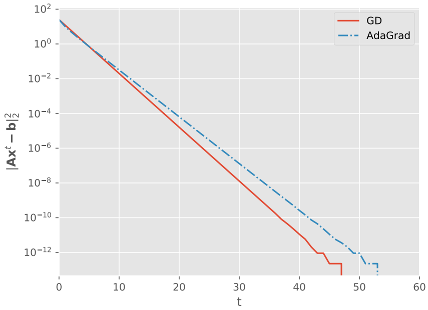


Consider the following simple 2D example :

$$\begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 5$$

Different Solutions

Gradient Descent and AdaGrad converge to *different* points on the line.



Implicit bias of gradient descent for linear regression

◦ Gradient descent seems to converge to the closest one in terms of ℓ_2-norm.

Theorem (Implicit bias of Gradient Descent [16])

For the underdetermined, realizable linear system

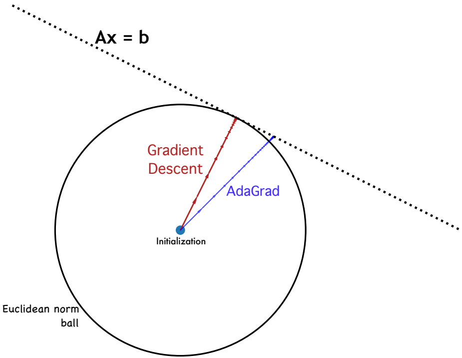
$$F^\star = \min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$$

the gradient descent algorithm $GD_{(T,\alpha,\mathbf{x}^0,\nabla F)}$*, for* $T = \infty$ *and for any* $\mathbf{x}^0 \in \mathbb{R}^p$*, and valid step-size* α*, has implicit bias* $H(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}^0\|_2$*, i.e.,*

$$GD_{(T=\infty,\alpha,\mathbf{x}^0,\nabla F)} = \arg\min_{F(\mathbf{x})=F^\star} \|\mathbf{x} - \mathbf{x}_0\|_2.$$

Remark: ◦ The theorem also holds for stochastic gradient descent, see [2].

Same problem and same initialization vs different algorithms and different solutions



Proof : For simplicity, take $\mathbf{x}_0 = 0$.

- The gradient of F is $\mathbf{A}^T(\mathbf{A}\mathbf{x} - b)$.
- This implies that $\forall \mathbf{x}$, $\nabla f(\mathbf{x}) \in \mathbf{colspan}(\mathbf{A}^T)$.

GD iterates stay in the rowspan

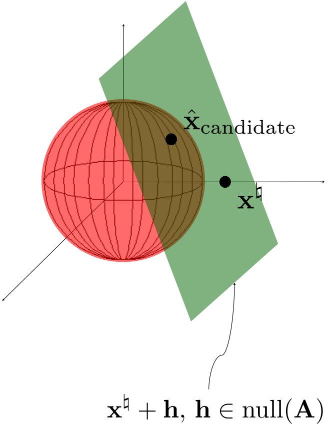
Gradient Descent is therefore constrained to the space

$$\mathbf{colspan}(\mathbf{A}^T) = \mathbf{rowspan}(\mathbf{A})$$

So its limit point at $T = \infty$ is in $\mathbf{rowspan}(\mathbf{A})$.

- Note that because of the preconditionning, AdaGrad can get out of the $\mathbf{rowspan}(\mathbf{A})$.

Same problem and same initialization vs different algorithms and different solutions



Proof (continued):

- The minimum norm solution

$$\hat{\mathbf{x}}_{\mathsf{candidate}} = \arg\min_{\mathbf{x}:\mathbf{A}\mathbf{x}=\mathbf{b}} \|\mathbf{x}\|_2^2$$

  is also in $\mathbf{rowspan}(\mathbf{A})$.
- So both $\hat{\mathbf{x}}_{\mathsf{candidate}}$ and the limit point of GD are solutions of $\mathbf{A}\mathbf{x} = \mathbf{b}$ that are in the $\mathbf{rowspan}(\mathbf{A})$
- Since $\mathbf{null}\mathbf{A} \cap \mathbf{rowspan}(\mathbf{A}) = \{0\}$, there can only be one solution in the $\mathbf{rowspan}(\mathbf{A})$, so

$$\mathbf{x}^\star_{\mathsf{GD}} = \hat{\mathbf{x}}_{\mathsf{candidate}}$$

Implicit bias for linear models

◦ We can extend this analysis to linear models:

$$\arg\min_{\mathbf{x}\in\mathbb{R}^p} F(\mathbf{x}) := \sum_{i=1}^{n} L(\langle \mathbf{x}, \mathbf{a}_i \rangle, b_i).$$

◦ If the observations are realizable and there are many global minima $\mathbf{Glob} = \{\mathbf{x} : F(\mathbf{x}) = 0\}$, then

Theorem (Implicit Bias of Gradient Descent [16])

If the loss L is convex and has a unique (attained) minimum, then the iterates $\mathbf{x}^t$ of Gradient Descent converge to the global minimum that is closest to initialization $\mathbf{x}_0$ in the ℓ_2-distance :

$$\mathbf{x}^t \xrightarrow[t\to\infty]{} \arg\min_{\mathbf{x}\in\mathbf{Glob}} \|\mathbf{x} - \mathbf{x}_0\|_2$$

Proof : (Sketch) The assumption on L implies the problem reduces to a linear system: If $\mathbf{x}$ is a global minimum, we must have $\langle \mathbf{x}, \mathbf{a}_i \rangle = b_i$ for all $i \in \{1, .., n\}$. We can recycle the results we have just seen.

Implicit bias for wide two-layer neural networks

- Assume a wide two-layer neural network $h_{\mathbf{x}}(\mathbf{a}) = \frac{1}{m}\sum_{i=1}^{m}\sigma(\langle \mathbf{x}_i, \mathbf{a}\rangle)$, where m is the width
- An integral representation parameterized with a probability measure ν is given by

$$h_\nu(\mathbf{a}) = \int_{\mathbb{R}^p} \sigma(\langle \mathbf{x}, \mathbf{a}\rangle)\mathrm{d}\nu(\mathbf{x})\,.$$

Theorem (Implicit bias of gradient flow on two-layer neural networks [12])

Under proper initialization and technical conditions (in particular, of convergence), the output of the gradient flow h_{ν_t} under a proper normalization scheme converges to a certain max-margin classifier.

Remarks:
- Gradient flow is the continuous limit of gradient descent [40].
- Fixing the hidden layer (i.e., random features) leads to a max-margin classifier in RKHS [12].
- Other extensions of implicit bias of SGD depend on different models or settings:
  - overparameterized least squares [43], diagonal linear networks [36], stochastic differential equations [23].
  - multi-pass SGD [44], diferent noise types [11, 18], different momentum types [42].

Double descent

◦ A failure of conventional wisdom

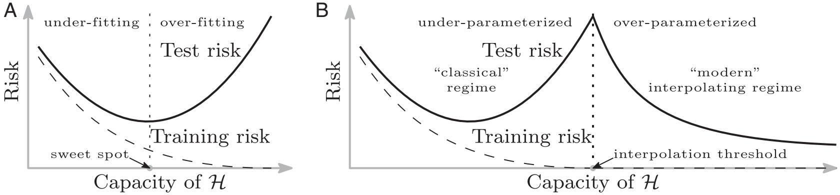


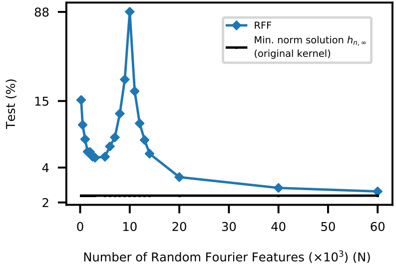
Figure: The classical U-shaped risk curve *vs.* double-descent risk curve. source: [10].

- classical large-sample limit setting: $n \to \infty$ under fixed p
- high dimensional setting: n and p comparably large

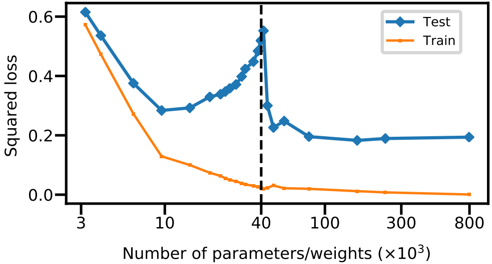
Double descent curve in practice (I)

Typical examples:

- linear/nonlinear regression [20]
- random features, random forest, and shallow neural networks [10]



(a) Random features model



(b) A fully connected neural network

Figure: Experiments on MNIST. Source: [10].

Double descent curve in practice (II)

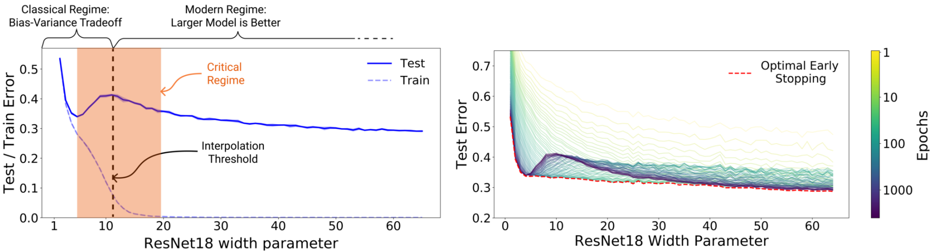


Figure: Left: Train and test error as a function of model size, for ResNet18s of varying width on CIFAR-10 with 15% label noise. Right: Test error, shown for varying train epochs. source: [31].

Double descent curve in practice (III)

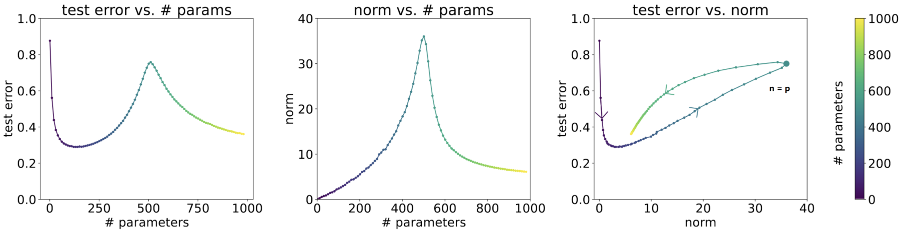


Figure: Left: The double descent phenomenon, where the number of parameters is used as the model complexity. Middle: The norm of the learned model is peaked around $n \approx p$. Right: The test error against the norm of the learnt model. The color bar indicate the number of parameters and the arrows indicates the direction of increasing model size. Their relationship are closer to the convention wisdom than to a double descent. source: [35]. This is the same setting as in Section 5.2 of [32].

Underparametrized regime

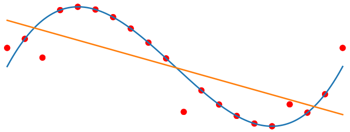


Figure: Low generalization but high empirical error

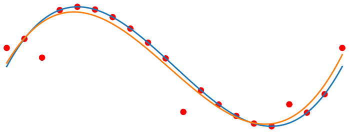


Figure: Sweet spot for the model complexity

Interpolation threshold

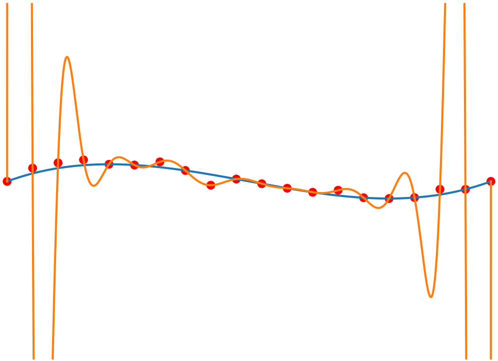


Figure: The unique degree 19 polynomial that can fit 20 samples.

Benign overfitting in the over-parametrized regime

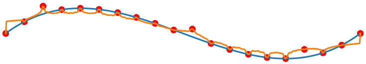


Figure: A degree 200 polynomial that can harmlessly fits noisy 20 points.

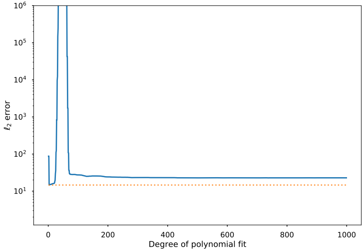


Figure: Double descent for polynomial fits

Benign Overfitting [8]: good prediction with zero training error for regression loss

- Statistical wisdom: a predictor should not fit too well.
- deep networks fit perfectly on noisy data and generalize well on test data.

A simple case: linear regression with Gaussian data

Problem setting

- linear model:
  - training data $\{\mathbf{a}_i\}_{i=1}^n$ with $\mathbf{a}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_p)$ such that $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_n]^\top$
  - label $b_i = \langle \mathbf{x}^\natural, \mathbf{a}_i \rangle + w_i$ with the target vector $\mathbf{x}^\natural \sim \mathcal{N}(\mathbf{0}, \frac{1}{p}\mathbf{I}_p)$, noise $w_i \sim \mathcal{N}(0, \sigma^2)$
- min-norm solution:
  - $\mathbf{x}^\star = \arg\min_{\mathbf{x}} \{\|\mathbf{x}\|_2 : \mathbf{A}\mathbf{x} = \mathbf{b}\} = (\mathbf{A}^\top \mathbf{A})^\dagger \mathbf{A}^\top \mathbf{b}$
- excess risk: for a test point $\mathbf{a}$
  - $R(\mathbf{x}^\star; \mathbf{x}^\natural) = \mathbb{E}[(\langle \mathbf{a}, \mathbf{x}^\star \rangle - \langle \mathbf{a}, \mathbf{x}^\natural \rangle)^2 | \mathbf{A}]$

Theorem [20]

Under the above problem setting, assume that $\|\mathbf{x}^\natural\|_2^2 = r^2$, as n, $p \to \infty$, and $p/n \to \gamma$, then we have

$$R(\mathbf{x}^\star, \mathbf{x}^\natural) \to \begin{cases} \sigma^2 \dfrac{\gamma}{1-\gamma}, & \text{for } \gamma < 1 \\ r^2 \left(1 - \dfrac{1}{\gamma}\right) + \sigma^2 \dfrac{1}{\gamma - 1}, & \text{for } \gamma > 1\,. \end{cases}$$

Remark: ◦ The asymptotic risk curves depend on γ and the SNR r^2/σ^2.

Double descent in 1998: AdaBoost

Definition (Informal [39])

"Boosting solves hard machine learning problems by forming a very smart committee of grossly incompetent but carefully selected members."

AdaBoost

1. Initialize the observation weights $w_i = 1/N$, $i = 1, 2, \ldots, N$
2. For $t = 1$ to T:
   2.1 Fit a classifier $h_{\mathbf{x},t}(\mathbf{a})$ to the training data using weights w_i.
   2.2 Compute
   $$\mathrm{err}_t = \frac{\sum_{i=1}^{N} w_i I(\mathbf{b}_i \neq h_{\mathbf{x},t}(\mathbf{a}_i))}{\sum_{i=1}^{N} w_i}.$$
   2.3 Compute $\alpha_t = \log((1 - \mathrm{err}_t)/\mathrm{err}_t)$.
   2.4 Set $w_i \leftarrow w_i \cdot \exp[\alpha_t \cdot I(\mathbf{b}_i \neq h_{\mathbf{x},t}(\mathbf{a}_i))]$, $i = 1, 2, \ldots, N$.
3. Output $h(\mathbf{a}) = \left[\sum_{t=1}^{T} \alpha_t h_{\mathbf{x},t}(\mathbf{a})\right]$.

Remarks:
- ◦ At each round, the weights are updated so the weak learner focuses on the hard examples.
- ◦ The more iterations are run, the more complex the output function becomes (e.g., overfitting).

AdaBoost with large number of rounds

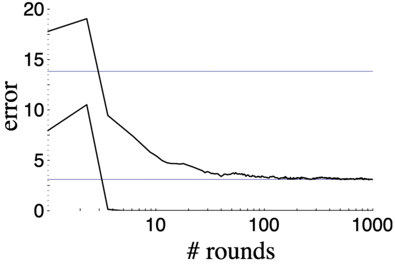
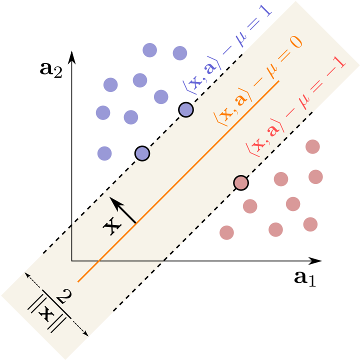


Figure: AdaBoost on letters dataset [6]. Test error keeps improving even after 0 training error is reached.



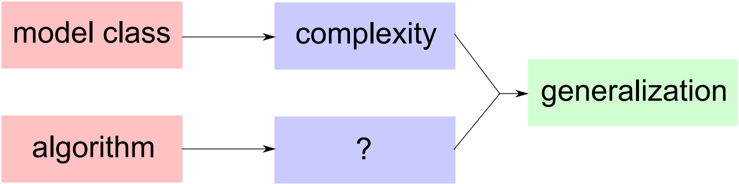
Margin theory [6]

The *margin* is a measure of confidence in the prediction. Boosting can be shown to increase the margin at each round.

Alternatives to complexity-based generalization bounds

- So far we have seen that complexity based generalization bounds:
  - characterize worst-case scenario
  - not tight in practice
  - disregard the effect of the optimization algorithm

Can we understand generalization as a property of an optimization algorithm?

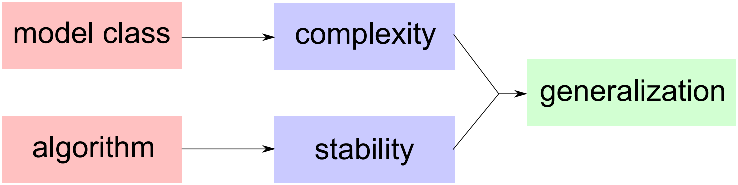


Alternatives to complexity-based generalization bounds

◦ So far we have seen that complexity based generalization bounds:

- characterize worst-case scenario
- not tight in practice
- disregard the effect of the optimization algorithm

Can we understand generalization as a property of an optimization algorithm? YES!



Formal definition of stability (I)

Definition (Uniform Stability [19])

Let $\mathscr{A} : \mathcal{Z} \to \mathcal{H}$ be a randomized algorithm with input a finite sample S, and output a function $\mathscr{A}_S \in \mathcal{H}$.

The algorithm $\mathscr{A}$ has uniform stability $(\beta_n)_{n\geq 1}$ with respect to the loss function L if for all subsets $S, S' \subseteq \mathcal{A} \times \mathcal{B}$ such that $|S| = |S'| = n$ and S and S' differ in at most one sample:

$$\sup_{(\mathbf{a},b)\in\mathcal{A}\times\mathcal{B}} \mathbb{E}|L(\mathscr{A}_S(\mathbf{a}), b) - L(\mathscr{A}_{S'}(\mathbf{a}), b)| \leq \beta_n$$

The expectation is taken with respect to the randomness in the algorithm $\mathscr{A}$.

Misnomer: Lower stability (small values of β_n) means the difference in the output of the algorithm is smaller.

Formal definition of stability (II)

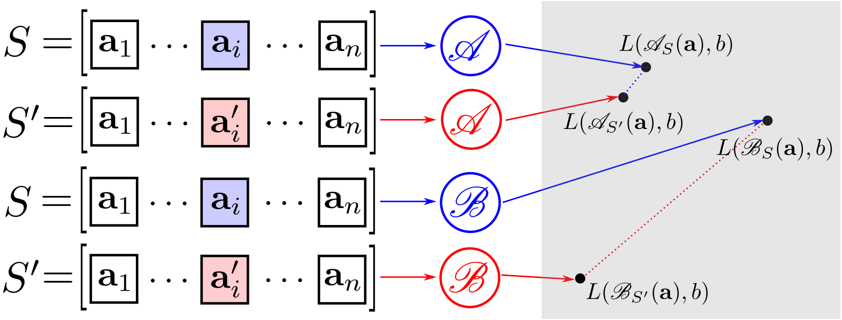


Figure: Algorithm $\mathscr{B}$ is less stable than algorithm $\mathscr{A}$.

The stability of SGD

- Let $h_{\mathbf{x}} \in \mathcal{H}_{\mathcal{X}}$ be an element of a parametric function class. Consider the ERM optimization objective:

$$f(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}), \qquad f_i(\mathbf{x}) := L(h_{\mathbf{x}}(\mathbf{a}_i), b_i).$$

- The SGD iterates for $t = 0, \ldots, T$ are $\mathbf{x}_{t+1} = \mathbf{x}_t - \alpha_t \nabla_{\mathbf{x}} f_i(\mathbf{x}_t)$, for $i \sim \mathrm{Unif}[n]$.

Algorithm	Assumptions on f_i	Stability
SGD	convex, L-smooth, β-Lipschitz, $\alpha_t \leq 2/L$	$\frac{\beta^2}{n}\sum_{t=0}^{T}\alpha_t$
SGD	μ-str convex, L-smooth, β-Lipschitz, $\alpha_t \leq 2/L$	$\frac{\beta^2}{n\mu}$
SGD	μ-str convex, L-smooth, β-Lipschitz, $\alpha_t = \frac{1}{\mu t}$	$\frac{\beta^2 + L\rho}{n\mu}$
SGD avg. iterate	convex, L-smooth, β-Lipschitz	$\frac{\beta^2 T}{nL}$
SGD	non-convex, L-smooth, β-Lipschitz, $\alpha_t = 1/t$	$\frac{1+1/\beta}{n}\beta^{\frac{2}{L+1}} T^{\frac{L}{L+1}}$

Table: Summary of stability upper bounds for different assumptions on the objective function [19]

Effect of the number of iterations on the stability of SGD and the generalization error

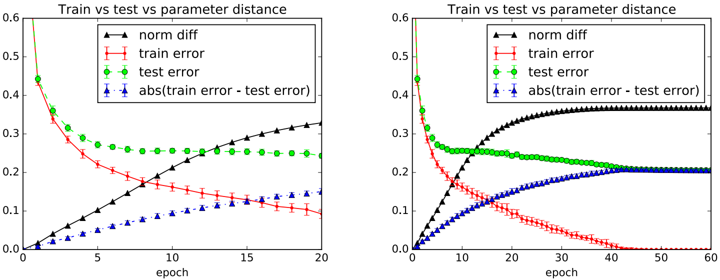


Figure: Normalized parameter distance between two networks trained on two datasets S, S' differing only in one sample, training error, test error and generalization error (0-1 loss) on CIFAR10 [19].

- Parameter distance is a stronger notion than stability.
- More iterations $\Rightarrow$ Parameter distance increases (we expect stability to increase).
- Generalization error follows the same behavior as the parameter distance (proxy for stability).

Wrap up!

- The visualizations can be deceiving to understand the high-dimensional behavior
- Are we really in the interpolation regime in machine learning?

Theorem (Probability of interpolation [5])

Given a p-dimensional dataset $\mathcal{A}_n = \{\mathbf{a}_1, \ldots, \mathbf{a}_n\}$ with i.i.d. samples, where $\mathbf{a}_i \sim \mathcal{N}(0, I)$ for all $i = 1, \ldots, n$, the probability that a new sample $\mathbf{a} \sim \mathcal{N}(0, I)$ is in the interpolation regime (i.e., within the convex hull of $\mathcal{A}_n$) has the following limiting behavior

$$\lim_{p \to \infty} p(\mathbf{a} \in \mathit{ConvexHull}(\mathcal{A}_n)) = \begin{cases} 1 & \textit{if } \; n > 2^{p/2}/p; \\ 0 & \textit{if } \; n < 2^{p/2}/p. \end{cases}$$

- We are most likely in the extrapolation regime [4]

*Concentration inequality

- Main tool for generalization bound: concentration inequalities!
  - Measure of how far is an empirical average from the true mean

Theorem (Hoeffding's Inequality [28])

Let $Y_1, \ldots, Y_n$ be i.i.d. random variables with Y_i taking values in the interval $[a_i, b_i] \subseteq \mathbb{R}$ for all $i = 1, \ldots, n$. Let $S_n := \frac{1}{n}\sum_{i=1}^n Y_i$. It holds that

$$\mathbb{P}\left(|S_n - \boldsymbol{E}[S_n]| > t\right) \leq 2\exp\left(-\frac{2n^2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right)$$

⋆Generalization bound for a singleton

Lemma

For $i = 1, \dots, n$, let $(\mathbf{a}_i, b_i) \in \mathbb{R}^p \times \{-1, 1\}$ be independent random variables and $h_{\mathbf{x}} : \mathbb{R}^p \to \mathbb{R}$ be a function parametrized by $\mathbf{x} \in \mathcal{X}$. Let $\mathcal{X} = \{\mathbf{x}_0\}$ and $L(h_{\mathbf{x}}(\mathbf{a}), b) = \{\text{sign}(h_{\mathbf{x}}(\mathbf{a})) \neq b\}$ be the 0-1 loss.
With probability at least $1 - \delta$, we have that

$$\sup_{\mathbf{x} \in \mathcal{X}} |R(\mathbf{x}) - R_n(\mathbf{x})| = |R(\mathbf{x}_0) - R_n(\mathbf{x}_0)| \leq \sqrt{\frac{\ln(2/\delta)}{2n}}.$$

Proof.

Note that $\boldsymbol{E}[\frac{1}{n}\sum_{i=1}^n L(h_{\mathbf{x}_0}(\mathbf{a}_i), b_i)] = R(\mathbf{x}_0)$, the expected risk of the parameter $\mathbf{x}_0$. Moreover $L(h_{\mathbf{x}_0}(\mathbf{a}_i), b_i) \in [0, 1]$. We can use Hoeffding's inequality and obtain

$$\mathbb{P}(|R_n(\mathbf{x}_0) - R(\mathbf{x}_0)| > t) = \mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^n L_i(h_{\mathbf{x}_0}(\mathbf{a}_i), b_i) - R(\mathbf{x}_0)\right| > t\right) \leq 2\exp\left(-2nt^2\right)$$

Setting $\delta := 2\exp\left(-2nt^2\right)$ we have that $t = \sqrt{\frac{\ln 2/\delta}{2n}}$, thus obtaining the result. □

⋆Generalization bound for finite sets

Lemma

For $i = 1, \ldots, n$, *let* $(\mathbf{a}_i, b_i) \in \mathbb{R}^p \times \{-1, 1\}$ *be independent random variables and* $h_{\mathbf{x}} : \mathbb{R}^p \to \mathbb{R}$ *be a function parametrized by* $\mathbf{x} \in \mathcal{X}$*. Let* $\mathcal{X}$ *be a finite set and* $L(h_{\mathbf{x}}(\mathbf{a}), b) = \{\mathit{sign}(h_{\mathbf{x}}(\mathbf{a})) \neq b\}$ *be the* 0-1 *loss.*
With probability at least $1 - \delta$, *we have that*

$$\sup_{\mathbf{x} \in \mathcal{X}} |R(\mathbf{x}) - R_n(\mathbf{x})| \leq \sqrt{\frac{\ln |\mathcal{X}| + \ln(2/\delta)}{2n}}.$$

Proof.

Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{|\mathcal{X}|}\}$. We can use a union bound and the analysis of the singleton case to obtain:

$$\mathbb{P}(\exists j : |R_n(\mathbf{x}_j) - R(\mathbf{x}_j)| > t) \leq \sum_{j=1}^{|\mathcal{X}|} \mathbb{P}(|R_n(\mathbf{x}_j) - R(\mathbf{x}_j)| > t) = 2|\mathcal{X}| \exp\Big(-2nt^2 \Big)$$

Setting $\delta := 2|\mathcal{X}| \exp\left(-2nt^2\right)$, we have that $t = \sqrt{\frac{\ln |\mathcal{X}| + \ln \frac{2}{\delta}}{2n}}$, thus obtaining the result. □

*Visualizing Rademacher complexity

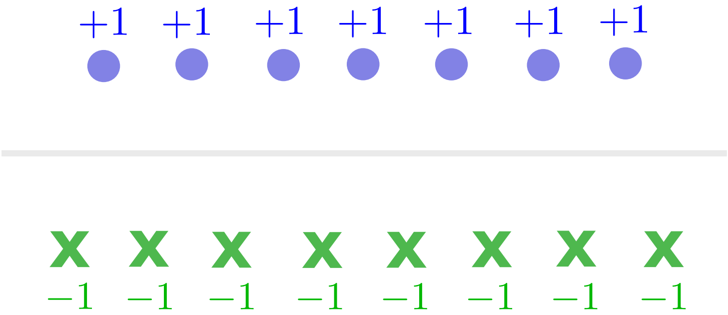
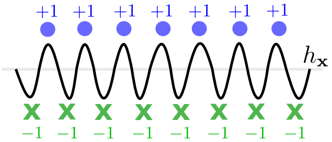
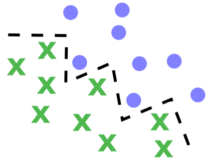


Figure: Rademacher complexity measures correlation with random signs

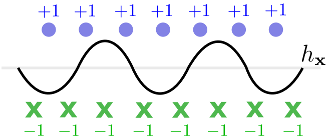
*Visualizing Rademacher complexity



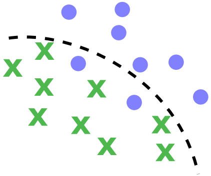
(a) High Rademacher Complexity



(b) Large Generalization error (memorization)



(c) Low Rademacher Complexity



(d) Low Generalization error

Figure: Rademacher complexity and Generalization error

*Computing the Rademacher complexity of linear functions

Theorem

Let $\mathcal{X} := \{\mathbf{x} \in \mathbb{R}^p : \|\mathbf{x}\|_2 \leq \lambda\}$ *and let* $\mathcal{H}_{\mathcal{X}}$ *be the class of functions of the form* $h_{\mathbf{x}} : \mathbb{R}^p \to \mathbb{R}, h_{\mathbf{x}}(\mathbf{a}) = \langle \mathbf{x}, \mathbf{a} \rangle$*, for some* $\mathbf{x} \in \mathcal{X}\}$*. Let* $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_n\} \subseteq \mathbb{R}^p$ *such that* $\max_{i=1,\ldots,n} \|\mathbf{a}_i\| \leq M$*. It holds that* $\mathcal{R}_A(\mathcal{H}_{\mathcal{X}}) \leq \lambda M / \sqrt{n}$*.*

Proof.

$$
\begin{aligned}
\mathcal{R}_A(\mathcal{H}_{\mathcal{X}}) &= \boldsymbol{E} \sup_{\|\mathbf{x}\|_2 \leq \lambda} \frac{1}{n} \sum_{i=1}^{n} v_i \langle \mathbf{x}, \mathbf{a} \rangle \\
&= \boldsymbol{E} \sup_{\|\mathbf{x}\|_2 \leq \lambda} \frac{1}{n} \left\langle \mathbf{x}, \sum_{i=1}^{n} v_i \mathbf{a} \right\rangle \\
&\leq \frac{1}{n} \lambda \boldsymbol{E} \left\| \sum_{i=1}^{n} v_i \mathbf{a}_i \right\|_2 \qquad \text{(C-S)}
\end{aligned}
$$

$$
\begin{aligned}
\Rightarrow \mathcal{R}_A(\mathcal{H}_{\mathcal{X}}) &\leq \frac{1}{n} \lambda \left(\boldsymbol{E} \sum_{i=1}^{n} \|v_i \mathbf{a}_i\|_2^2 \right)^{1/2} \qquad \text{(Jensen)} \\
&\leq \frac{1}{n} \lambda \left(\sum_{i=1}^{n} \|\mathbf{a}_i\|_2^2 \right)^{1/2} \\
&\leq \lambda M / \sqrt{n}
\end{aligned}
$$

□

*Rademacher complexity estimates of fully connected Neural Networks

Notation

For a matrix $\mathbf{X} \in \mathbb{R}^{n,m}$, $\|\mathbf{X}\|$ denotes its spectral norm. Let $\mathbf{X}_{:,k}$ be the k-th column of $\mathbf{X}$. We define

$$\|\mathbf{X}\|_{2,1} = \|(\|\mathbf{X}_{:,1}\|_2, \ldots, \|\mathbf{X}_{:,m}\|_2)\|_1. \tag{2}$$

Theorem (Spectral bound [7])

For positive integers $p_0, p_1, \ldots, p_d = 1$, and positive reals $\lambda_1, \ldots, \lambda_d$ and $\nu_1, \ldots, \nu_d$, define the set

$$\mathcal{X} := \{(\mathbf{X}_1, \ldots, \mathbf{X}_d) : \mathbf{X}_i \in \mathbb{R}^{p_i \times p_{i-1}}, \|\mathbf{X}_i\| \leq \lambda_i, \|\mathbf{X}_i^T\|_{2,1} \leq \nu_i\}$$

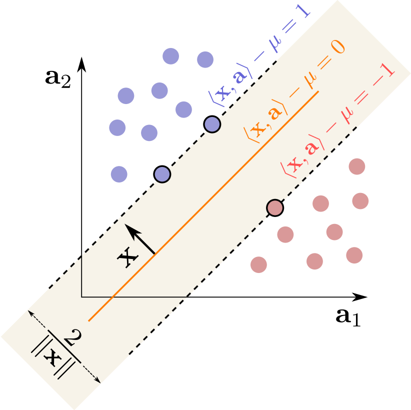
Let $H_{\mathcal{X}}$ be the class of neural networks $h_{\mathbf{x}} : \mathbb{R}^p \to \mathbb{R}$, $h_{\mathbf{x}} = \mathbf{X}_d \circ \sigma \circ \ldots \circ \sigma \circ \mathbf{X}_1$ where $\mathbf{x} = (\mathbf{X}_1, \ldots, \mathbf{X}_d) \in \mathcal{X}$. Suppose that σ is 1-Lipschitz. Let $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_n\} \subseteq \mathbb{R}^p$, $M := \max_{i=1,\ldots,n} \|\mathbf{a}_i\|$ and $W := \max\{p_i : i = 0, \ldots, d\}$.

The Rademacher complexity of $\mathcal{H}_{\mathcal{X}}$ with respect to A is bounded as

$$\mathcal{R}_A(\mathcal{H}_{\mathcal{X}}) = \mathcal{O}\left(\frac{\log(W)M}{\sqrt{n}} \prod_{i=1}^{d} \lambda_i \left(\sum_{j=1}^{d} \frac{\nu_j^{2/3}}{\lambda_j^{2/3}} \right)^{3/2} \right). \tag{3}$$

*Implicit bias for linearly separable datasets

- For linearly separable datasets, we know of an algorithm capable of finding a separating hyperplane.
- It maximizes the *margin* (i.e., distance between the boundary and the nearest training-data point).



Hard-margin Support Vector Machines

The hard margin Support Vector Machine solves the following optimization problem :

$$\arg\min_{\mathbf{x}\in\mathbb{R}^p} \|\mathbf{x}\|_2 \quad \text{subject to } y_i\langle \mathbf{x}, \mathbf{a}_i\rangle \geq 1.$$

It finds a hyperplane that maximizes the margin. It does so *by design*.

*Implicit bias for linearly separable datasets

◦ What happens if we do not explicitly enforce margin maximization?

Theorem (Implicit Bias of Gradient Descent on Separable Data [41, 16])

For the logistic loss (and some other strictly monotonically decreasing losses) and for linearly separable datasets, the direction of the iterates $\mathbf{x}^t$ of Gradient Descent for any initialization converges to the hard-margin SVM direction:

$$\frac{\mathbf{x}^t}{\|\mathbf{x}^t\|_2} \xrightarrow[t\to\infty]{} \frac{\mathbf{x}^\star_{SVM}}{\|\mathbf{x}^\star_{SVM}\|_2} \quad \text{where } \mathbf{x}^\star_{SVM} = \left\{ \arg\min_{\mathbf{x}\in\mathbb{R}^p} \|\mathbf{x}\|_2 \quad \text{subject to } y_i\langle \mathbf{x}, \mathbf{a}_i\rangle \geq 1 \right\}$$

Remarks:

◦ Here, without explicit instructions, gradient descent maximizes the margin.

◦ The rate of this convergence is $O\left(\frac{1}{\log t}\right)$.

*Implicit bias for linearly separable datasets

◦ A similar result can be established for stochastic gradient descent for the logistic loss on separable datasets.

Theorem (Implicit Bias of *Stochastic* Gradient Descent on Separable Data [29])

The direction of the iterates $\mathbf{x}^t$ of Stochastic Gradient Descent for any initialization and for a small enough fixed step-size, converges almost surely to the hard-margin SVM direction:

$$\left\| \frac{\mathbf{x}^t}{\|\mathbf{x}^t\|_2} - \frac{\mathbf{x}^\star_{SVM}}{\|\mathbf{x}^\star_{SVM}\|_2} \right\|_2 = O\left(\frac{1}{\log t}\right)$$

Remarks:

◦ This result is particularly interesting as it establishes convergence of fixed step-size SGD.

◦ Both SGD and GD have the same implicit bias towards maximizing margins.

*Implicit bias for non-convex objectives

◦ Characterizing implicit bias of stochastic gradient descent for non-convex objectives is an active research area.

◦ Some papers study deep matrix factorization as a first step towards getting results for neural networks.

Deep Matrix Factorization

Deep matrix factorization consists of parametrizing a matrix $\mathbf{M}$ as a product of N matrices:

$$\mathbf{M} = \mathbf{X}_N \mathbf{X}_{N-1} \ldots \mathbf{X}_1$$

which can be understood as parametrizing $\mathbf{M}$ by a depth N "*linear neural network*," i.e., a neural network with no activations and with weight matrices $\mathbf{X}$.

*Implicit bias for deep matrix completion

- The matrix completion problem consists of filling the missing entries of a partially observed matrix.
- The deep matrix factorization approach consists of solving the following problem with gradient descent:

$$\operatorname*{arg\,min}_{\mathbf{X}_N, \mathbf{X}_{N-1} \ldots, \mathbf{X}_1} \sum_{(i,j)\in\Omega} \left([\mathbf{X}_N \mathbf{X}_{N-1} \ldots \mathbf{X}_1]_{i,j} - b_{i,j}\right)^2.$$

- It was conjectured in 2017 [17] that gradient descent was biased towards solutions with small nuclear norm.

Theorem (Implicit Regularization May Not Be Explainable by Norms (2020) [38])

For deep matrix completion the implicit bias can not be expressed as a function of a norm or semi-norm.

⋆Implicit bias for wide two-layer neural networks

- Assume that we have a wide two-layer neural network $h_{\mathbf{x}}(\mathbf{a}) = \frac{1}{m}\sum_{i=1}^m \sigma(\langle \mathbf{x}_i, \mathbf{a}\rangle)$
- An integral representation parameterized with a probability measure ν

$$h_\nu(\mathbf{a}_i) = \int_{\mathbb{R}^p} \sigma(\langle \mathbf{x}, \mathbf{a}_i\rangle)\mathrm{d}\nu(\mathbf{x})\,,$$

- $\nu \in \mathcal{P}_2(\mathbb{R}^{d+2})$ in the set of probability measures with finite second moment
- the variation norm: $\|h\|_{\mathcal{F}_1} = \min_{\nu\in\mathcal{P}_2(\mathbb{R}^{d+2})} \left\{ \frac{1}{2}\int \|\mathbf{x}\|_2^2\,\mathrm{d}\nu(\mathbf{x});\quad h_\nu(\mathbf{a}_i) = \int \sigma(\langle \mathbf{x}, \mathbf{a}_i\rangle)\mathrm{d}\nu(\mathbf{x})\right\}$

Theorem (Implicit Bias of wide two-layer Neural Networks [12])

Assume that $\nu_0 = \mathcal{U}_{\mathbb{S}^d} \otimes \mathcal{U}_{\{-1,1\}}$, the training set is consistent ($[\mathbf{a}_i = \mathbf{a}_j] \implies [b_i = b_j]$) and technical conditions (in particular, of convergence). Then $h_{\nu_t} / \|h_{\nu_t}\|_{\mathcal{F}_1}$ trained by an exponential tail loss converges to the $\mathcal{F}_1$-max-margin classifier, i.e. it solves

$$\max_{\|h\|_{\mathcal{F}_1}\le 1} \min_{i\in[n]} b_i h(\mathbf{a}_i)\,,$$

- Gradient flow is the continuous limit of gradient descent [40].
- Fixing the hidden layer (i.e., random features) leads to a max-margin classifier in RKHS [12].

*From neural networks to random features model [21, 37]

1-hidden-layer neural network with m neurons (fully-connected architecture):

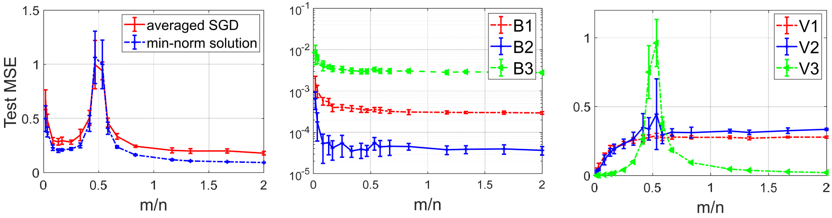
Let $\mathbf{X}_1 \in \mathbb{R}^{m\times p}$, $\mathbf{a} \in \mathbb{R}^p$, $\mathbf{X}_2 \in \mathbb{R}^m$, and $\mu_2 \in \mathbb{R}$

$$h_{\mathbf{x}}(\mathbf{a}) := \begin{bmatrix} \mathbf{X}_2 \end{bmatrix} \underbrace{\overset{\text{activation}}{\overset{\downarrow}{\sigma}} \left(\begin{bmatrix} \overset{\text{weight}}{\overset{\downarrow}{\mathbf{X}_1}} \end{bmatrix} \overset{\text{input}}{\overset{\downarrow}{\begin{bmatrix} \mathbf{a} \end{bmatrix}}} + \overset{\text{bias}}{\overset{\downarrow}{\begin{bmatrix} \mu_1 \end{bmatrix}}} \right)}_{\text{hidden layer} \,=\, \textit{fixed random features}} + \overset{\text{bias}}{\overset{\downarrow}{\begin{bmatrix} \mu_2 \end{bmatrix}}}, \qquad \mathbf{x} := [\mathbf{X}_1, \mathbf{X}_2, \mu_1, \mu_2]$$

- $\mathbf{X}_1$: Gaussian initialization and then fixed
- $\mathbf{X}_2$: to be learned
- over-parameterized model: #neurons $m >$ #training data n

*Double descent: random features model (I)

○ high dimensions: #training data n, #neurons m, feature dimension p are comparably large



(a) SGD vs. min-norm solution (b) Bias $\lesssim$ B1 + B2 + B3 (c) Variance $\lesssim$ V1 + V2 + V3

Figure: Test MSE, Bias, and Variance of RF regression as a function of the ratio m/n on MNIST data set (digit 3 vs. 7) for $p = 784$ and $n = 600$ across the Gaussian kernel. Source: [26].

- random features regression solved by SGD: interplay between excess risk and optimization
- bias variance decomposition for understanding multiple randomness sources
- monotonic decreasing bias and unimodal variance $\Rightarrow$ double descent

*Double descent: random features model (II)

Algorithm	data assumption	solution type	Result on risk curve
[20]	Gaussian	closed-form	variance $\nearrow \searrow$
[27]	i.i.d on sphere	closed-form	variance, bias $\nearrow \searrow$
[13]	Gaussian	closed-form	refined decomposition on variance
[1]	Gaussian	closed-form	fully decomposition on variance
[25]	general	closed-form	$\nearrow \searrow$
[3]	Gaussian	GD	variance $\nearrow \searrow$
[26]	sub-Gaussian	SGD	variance $\nearrow \searrow$, bias $\searrow$

Table: Comparison of representative random features on double descent.

- ◦ multiple randomness sources: data sampling, label noise, initialization
- ◦ phase transition due to non-monotonic variance

*Peeling the onion (risk minimization setting) - Decomposition details

$$R(\mathbf{x}^t) - R(\mathbf{x}^\natural) = R(\mathbf{x}^t) - R_n(\mathbf{x}^t) + R_n(\mathbf{x}^t) - R_n(\mathbf{x}^\star) + \underbrace{R_n(\mathbf{x}^\star) - R_n(\mathbf{x}^\natural)}_{\leq 0} + R_n(\mathbf{x}^\natural) - R(\mathbf{x}^\natural)$$

$$\leq R_n(\mathbf{x}^t) - R_n(\mathbf{x}^\star) + \underbrace{R(\mathbf{x}^t) - R_n(\mathbf{x}^t) + R_n(\mathbf{x}^\natural) - R(\mathbf{x}^\natural)}_{2\sup_{\mathbf{x}\in\mathcal{X}} |R_n(\mathbf{x}) - R(\mathbf{x})|}$$

$$\begin{aligned} R(\mathbf{x}^t) - R(\mathbf{x}^\circ) &= R(\mathbf{x}^t) - R(\mathbf{x}^\natural) + R(\mathbf{x}^\natural) - R(\mathbf{x}^\circ) \\ &\leq R_n(\mathbf{x}^t) - R_n(\mathbf{x}^\star) + 2\sup_{\mathbf{x}\in\mathcal{X}} |R_n(\mathbf{x}) - R(\mathbf{x})| + R(\mathbf{x}^\natural) - R(\mathbf{x}^\circ) \end{aligned}$$

*Generalization bounds based on uniform stability — definitions

Definition (Empirical Risk on a set)

Let $S := [(\mathbf{a}_1, b_1), \ldots, (\mathbf{a}_n, b_n)]$ be an i.i.d. sample drawn from a distribution on $\mathcal{A} \times \mathcal{B}$. Let $L : \mathcal{B} \times \mathcal{B} \to \mathbb{R}$ be a loss function and $\mathcal{H}$ be a class of functions $h : \mathcal{A} \to \mathcal{B}$. The empirical risk of $h \in \mathcal{H}$ on the set S is defined as:

$$R_S(h) := \frac{1}{n} \sum_{i=1}^{n} L(h(\mathbf{a}_i), b_i)$$

(Almost) same definition as before. Makes explicit the dependence on the set S.

Definition (Expected Generalization Error)

Let $\mathscr{A} : \mathcal{Z} \to \mathcal{H}$ be a randomized algorithm that takes as input a finite sample S of arbitrary size, and outputs a function $\mathscr{A}_S \in \mathcal{H}$. Suppose that $S = [(\mathbf{a}_1, b_1), \ldots, (\mathbf{a}_n, b_n)]$ is an i.i.d. sample form probability distribution on $\mathcal{A} \times \mathcal{B}$. The expected generalization error on a sample of size n is the value

$$\mathbb{E}[R_S(\mathscr{A}_S) - R(\mathscr{A}_S))]$$

the expectation is taken with respect to the draw of the sample S and the randomness of $\mathscr{A}$.

*Generalization bounds based on uniform stability — Fundamental theorem (I)

Theorem (Hardt et al. 2016 [19])

Let A be uniformly stable with stability $(\beta_n)_{n\geq 1}$, then for a random i.i.d. sample S of size n, the expected generalization error is bounded as follows

$$\mathbb{E}[|R_S(\mathscr{A}_S) - R(\mathscr{A}_S))|] \leq \beta_n$$

Proof.

Let $S = [(\mathbf{a}_1, b_1), \ldots, (\mathbf{a}_n, b_n)]$ and $S' = [(\mathbf{a}'_1, b'_1), \ldots, (\mathbf{a}'_n, b_n)]$ be two i.i.d. samples of size n. Denote

$$S^{(i)} := [(\mathbf{a}_1, b_1), \ldots, (\mathbf{a}_{i-1}, b_{i-1}), (\mathbf{a}'_i, b'_i), (\mathbf{a}_{i+1}, b_{i+1}), \ldots, (\mathbf{a}_n, b_n)]$$

the sample that results from replacing $(\mathbf{a}_i, b_i)$ by $(\mathbf{a}'_i, b'_i)$ in S.

$$\begin{aligned}
\mathbb{E}[R_S(\mathscr{A}_S)] &= \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^n L(\mathscr{A}_S(\mathbf{a}_i), b_i)\right] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^n L(\mathscr{A}_{S^{(i)}}(\mathbf{a}'_i), b'_i)\right] \\
&= \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^n L(\mathscr{A}_{S^{(i)}}(\mathbf{a}'_i), b'_i) - \frac{1}{n}\sum_{i=1}^n L(\mathscr{A}_S(\mathbf{a}'_i), b'_i)\right] + \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^n L(\mathscr{A}_S(\mathbf{a}'_i), b'_i)\right]
\end{aligned}$$

$\square$

⋆Generalization bounds based on uniform stability — Fundamental theorem (II)

Proof. (continued).

We have

$$\mathbb{E}[R_S(\mathscr{A}_S)] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L(\mathscr{A}_{S^{(i)}}(\mathbf{a}_i'), b_i') - \frac{1}{n}\sum_{i=1}^{n} L(\mathscr{A}_S(\mathbf{a}_i'), b_i')\right] + \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L(\mathscr{A}_S(\mathbf{a}_i'), b_i')\right]$$

Note that S and $S^{(i)}$ only differ in one sample: uniform stability allows bounding the first term as:

$$= \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L(\mathscr{A}_{S^{(i)}}(\mathbf{a}_i'), b_i') - \frac{1}{n}\sum_{i=1}^{n} L(\mathscr{A}_S(\mathbf{a}_i'), b_i')\right] = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[L(\mathscr{A}_{S^{(i)}}(\mathbf{a}_i'), b_i') - L(\mathscr{A}_S(\mathbf{a}_i'), b_i')\right] \leq \beta_n$$

Finally note that because the samples $(\mathbf{a}_i, b_i)$ are independent of S we have:

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L(\mathscr{A}_S(\mathbf{a}_i'), b_i')\right] = R(\mathscr{A}_S)$$

analogously we can show $\mathbb{E}\left[R(\mathscr{A}_S) - R_S(\mathscr{A}_S)\right] \leq \beta_n$. □

References I

[1] Ben Adlam and Jeffrey Pennington.
Understanding double descent requires a fine-grained bias-variance decomposition.
In *Advances in neural information processing systems*, 2020.
(Cited on page 65.)

[2] Navid Azizan and Babak Hassibi.
Stochastic gradient/mirror descent: Minimax optimality and implicit regularization.
(Cited on page 28.)

[3] Jimmy Ba, Murat A. Erdogdu, Taiji Suzuki, Denny Wu, and Tianzong Zhang.
Generalization of two-layer neural networks: an asymptotic viewpoint.
In *International Conference on Learning Representations*, pages 1–8, 2020.
(Cited on page 65.)

[4] Randall Balestriero, Jerome Pesenti, and Yann LeCun.
Learning in high dimension always amounts to extrapolation.
arXiv preprint arXiv:2110.09485, 2021.
(Cited on page 49.)

References II

[5] Imre Bárány and Zoltán Füredi.
On the shape of the convex hull of random points.
Probability theory and related fields, 77(2):231–240, 1988.
(Cited on page 49.)

[6] Peter Bartlett, Yoav Freund, Wee Sun Lee, and Robert E Schapire.
Boosting the margin: A new explanation for the effectiveness of voting methods.
The annals of statistics, 26(5):1651–1686, 1998.
(Cited on page 42.)

[7] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky.
Spectrally-normalized margin bounds for neural networks.
In *Advances in Neural Information Processing Systems 30*, pages 6240–6249. Curran Associates, Inc., 2017.
(Cited on pages 16 and 56.)

[8] Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler.
Benign overfitting in linear regression.
Proceedings of the National Academy of Sciences, 117(48):30063–30070, 2020.
(Cited on page 39.)

References III

[9] Peter L Bartlett and Shahar Mendelson.
Rademacher and gaussian complexities: Risk bounds and structural results.
Journal of Machine Learning Research, 3(Nov):463–482, 2002.
(Cited on page 12.)

[10] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal.
Reconciling modern machine-learning practice and the classical bias–variance trade-off.
Proceedings of the National Academy of Sciences, 116(32):15849–15854, 2019.
(Cited on pages 33 and 34.)

[11] Alexander Camuto, Xiaoyu Wang, Lingjiong Zhu, Chris Holmes, Mert Gurbuzbalaban, and Umut Simsekli.
Asymmetric heavy tails and implicit bias in gaussian noise injections.
In *Proceedings of the 38th International Conference on Machine Learning*, 2021.
(Cited on page 32.)

[12] Lenaic Chizat and Francis Bach.
Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss.
In *Conference on Learning Theory*, 2020.
(Cited on pages 32 and 62.)

References IV

[13] Stéphane d'Ascoli, Maria Refinetti, Giulio Biroli, and Florent Krzakala.
Double trouble in double descent: Bias and variance (s) in the lazy regime.
pages 2280–2290, 2020.
(Cited on page 65.)

[14] Gintare Karolina Dziugaite, Alexandre Drouin, Brady Neal, Nitarshan Rajkumar, Ethan Caballero, Linbo Wang, Ioannis Mitliagkas, and Daniel M Roy.
In search of robust measures of generalization.
Advances in Neural Information Processing Systems, 33:11723–11733, 2020.
(Cited on page 16.)

[15] Stuart Geman, Elie Bienenstock, and René Doursat.
Neural networks and the bias/variance dilemma.
Neural computation, 4(1):1–58, 1992.
(Cited on page 6.)

[16] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro.
Characterizing implicit bias in terms of optimization geometry.
(Cited on pages 28, 31, and 58.)

References V

[17] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro.
Implicit regularization in matrix factorization.
In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6151–6159. Curran Associates, Inc.
(Cited on page 61.)

[18] Jeff Z. HaoChen, Colin Wei, Jason Lee, and Tengyu Ma.
Shape matters: Understanding the implicit bias of the noise covariance.
In *Proceedings of Thirty Fourth Conference on Learning Theory*, 2021.
(Cited on page 32.)

[19] Moritz Hardt, Ben Recht, and Yoram Singer.
Train faster, generalize better: Stability of stochastic gradient descent.
In *International Conference on Machine Learning*, pages 1225–1234. PMLR, 2016.
(Cited on pages 45, 47, 48, and 68.)

[20] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani.
Surprises in high-dimensional ridgeless least squares interpolation.
arXiv preprint arXiv:1903.08560, 2019.
(Cited on pages 34, 40, and 65.)

References VI

[21] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew.
Extreme learning machine: theory and applications.
Neurocomputing, 70(1-3):489–501, 2006.
(Cited on page 63.)

[22] Yiding Jiang*, Behnam Neyshabur*, Hossein Mobahi, Dilip Krishnan, and Samy Bengio.
Fantastic generalization measures and where to find them.
In *International Conference on Learning Representations*, 2020.
(Cited on page 16.)

[23] Zhiyuan Li, Tianhao Wang, and Sanjeev Arora.
What happens after SGD reaches zero loss? –a mathematical framework.
In *International Conference on Learning Representations*, 2022.
(Cited on page 32.)

[24] Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes.
Fisher-rao metric, geometry, and complexity of neural networks.
volume 89 of *Proceedings of Machine Learning Research*, pages 888–896. PMLR, 16–18 Apr 2019.
(Cited on page 16.)

References VII

[25] Zhenyu Liao, Romain Couillet, and Michael Mahoney.
A random matrix analysis of random fourier features: beyond the gaussian kernel, a precise phase transition, and the corresponding double descent.
In *Neural Information Processing Systems*, 2020.
(Cited on page 65.)

[26] Fanghui Liu, Johan A.K. Suykens, and Volkan Cevher.
On the double descent of random features models trained with sgd.
arXiv preprint arXiv:2110.06910, 2021.
(Cited on pages 64 and 65.)

[27] Song Mei and Andrea Montanari.
The generalization error of random features regression: Precise asymptotics and double descent curve.
arXiv preprint arXiv:1908.05355, 2019.
(Cited on page 65.)

[28] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar.
Foundations of Machine Learning.
The MIT Press, 2nd edition, 2018.
(Cited on pages 13 and 50.)

References VIII

[29] Mor Shpigel Nacson, Nathan Srebro, and Daniel Soudry.
Stochastic gradient descent on separable data: Exact convergence with a fixed learning rate.
(Cited on page 59.)

[30] Vaishnavh Nagarajan and J. Zico Kolter.
Generalization in Deep Networks: The Role of Distance from Initialization.
arXiv e-prints, page arXiv:1901.01672, January 2019.
(Cited on page 16.)

[31] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever.
Deep double descent: Where bigger models and more data hurt.
(Cited on page 35.)

[32] Preetum Nakkiran, Prayaag Venkat, Sham M. Kakade, and Tengyu Ma.
Optimal regularization can mitigate double descent.
In *International Conference on Learning Representations*, 2021.
(Cited on page 36.)

[33] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro.
The role of over-parametrization in generalization of neural networks.
In *International Conference on Learning Representations*, 2019.
(Cited on pages 14 and 15.)

References IX

[34] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro.
Norm-based capacity control in neural networks.
In *Conference on Learning Theory*, pages 1376–1401, 2015.
(Cited on page 16.)

[35] Andrew Ng.
Cs229 lecture notes, 2022.
(Cited on page 36.)

[36] Scott Pesme, Loucas Pillaud-Vivien, and Nicolas Flammarion.
Implicit bias of SGD for diagonal linear networks: a provable benefit of stochasticity.
In *Advances in Neural Information Processing Systems*, 2021.
(Cited on page 32.)

[37] Ali Rahimi and Benjamin Recht.
Random features for large-scale kernel machines.
In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2007.
(Cited on page 63.)

[38] Noam Razin and Nadav Cohen.
Implicit regularization in deep learning may not be explainable by norms.
(Cited on page 61.)

References X

[39] Robert E Schapire and Yoav Freund.
Boosting: Foundations and algorithms.
Kybernetes, 2013.
(Cited on page 41.)

[40] Damien Scieur, Vincent Roulet, Francis Bach, and Alexandre d Aspremont.
Integration methods and optimization algorithms.
In *Advances in Neural Information Processing Systems*, 2017.
(Cited on pages 32 and 62.)

[41] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro.
The implicit bias of gradient descent on separable data.
(Cited on page 58.)

[42] Bohan Wang, Qi Meng, Huishuai Zhang, Ruoyu Sun, Wei Chen, Zhi-Ming Ma, and Tie-Yan Liu.
Does momentum change the implicit regularization on separable data?
In *Advances in Neural Information Processing Systems*, 2022.
(Cited on page 32.)

References XI

[43] Jingfeng Wu, Difan Zou, Vladimir Braverman, and Quanquan Gu.
Direction matters: On the implicit bias of stochastic gradient descent with moderate learning rate.
In *International Conference on Learning Representations*, 2021.
(Cited on page 32.)

[44] Difan Zou, Jingfeng Wu, Vladimir Braverman, Quanquan Gu, and Sham M Kakade.
Risk bounds of multi-pass sgd for least squares in the interpolation regime.
In *Advances in Neural Information Processing Systems*, 2022.
(Cited on page 32.)