Statistical Learning Theory: Old and New

Ulrike von Luxburg
(University of Tübingen)

MLSS 2026



Contents

Lecture 1: Classical Statistical Learning Theory

Lecture 2: Double descent and implicit regularization

Lecture 3: Benign overfitting

Lecture 1: Classical theory

Literature:

- Bousquet, Olivier, Stephane Boucheron, and Gabor Lugosi: Introduction to statistical learning theory. These are part of the proceedings of the very first MLSS in Tübingen, 2003.
- Shalev-Shwartz and Ben-David: Understanding machine learning, 2014.
- Francis Bach: Learning theory from first principles, 2024.

Setup and main approach

Statistical setup

- Input space $\mathcal{X}$, output space $\mathcal{Y}$
- Probability distribution P on the product space $\mathcal{X} \times \mathcal{Y}$
- A classifier or a prediction function is a function $f : \mathcal{X} \to \mathcal{Y}$.

How good is a particular classifier?

- We use a loss function to measure how "expensive" an error is: el $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$.
- The true risk of a prediction function $f : \mathcal{X} \to \mathcal{Y}$ is defined as

$$R(f) := E\big(\ell(Y, f(X))\big)$$

where the expectation is over the random draw of (X, Y) according to the probability distribution P on $\mathcal{X} \times \mathcal{Y}$.

Statistical setup (2)

What is the best possible classifier?

- The Bayes risk is defined as

$$R^* := \inf\{R(f) \mid f : \mathcal{X} \to \mathcal{Y},\ f \text{ measurable } \}$$

- In case the infimum is attained, the corresponding function

$$f^* := \operatorname{argmin} R(f)$$

  is called the Bayes classifier / Bayes predictor.

Machine learning

- We get training data by the following process:
  We draw n training points $(X_i, Y_i)_{i=1,\dots,n} \in \mathcal{X} \times \mathcal{Y}$ i.i.d. (independent and identically distributed) according to probability distribution P.
- Given n training points, **the goal of learning is to construct a function f_n that has true risk "close" to the Bayes risk, that is $R(f_n) \approx R^*$.**
- If we knew P, we could try to run some procedure that finds a function with small true risk $R(f_n)$. But we don't know P, so we need to use some "proxy".
- The obvious candidate is the empirical risk:

$$R_n(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(X_i), Y_i)$$

- The challenge of machine learning is to devise a procedure that uses the empirical risk of functions to find one function f_n that – hopefully! – has a small true risk.
- The field of statistical learning theory tries to figure out when, under which assumptions, which kind of procedures can be successful in principle or not.

Consistency of a learning algorithm

- We say that a learning algorithm $\mathcal{A}$ is consistent for probability distribution P if the true risk $R(f_n)$ of its selected functions f_n converges to the Bayes risk as $n \to \infty$:

$$\forall \varepsilon > 0 : \lim_{n \to \infty} P(R(f_n) - R^* > \varepsilon) = 0.$$

- We say that algorithm $\mathcal{A}$ is universally consistent if it is consistent for all possible probability distributions P over $\mathcal{X} \times \mathcal{Y}$.

In words: No matter what the underlying probability distribution is, when we have seen "enough data points", then the true risk of our selected classifier f_n will be arbitrarily close to the best possible one.

Empirical risk minimization (ERM)

In the classical setup of learning theory, we study a theoretical "algorithm":

- We fix a set $\mathcal{F}$ of "candidate functions" from $\mathcal{X} \to \mathcal{Y}$. It could be the set of all linear functions, the set of all decision trees of depth 5, the set of all random forests, the set of all two-layer MLP, etc.
- We assume that within these functions, we select one that minimizes the empirical risk:

$$f_n := \operatorname*{argmin}_{f \in \mathcal{F}} R_n(f)$$

We now want to analyze whether / under which assumptions this "algorithm" is consistent.

How close are true and empirical risk of a fixed function f_0?

Given a function f_0 fixed in advance. Then draw training data. How are the two risks related?

$$R(f) = E\big(\ell(f(X), Y)$$

$$R_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(X_i), Y_i)$$

How close are true and empirical risk of a fixed function f_0? (2)

Proposition 1 (Law of large numbers and Hoeffding inequality)

1. Let $(Z_i)_{i\in\mathbb{N}}$ be a sequence of random variables that have been drawn independently according to some probability distribution P, denote their expectation as $E(Z)$. Then (under mild assumptions)

$$\frac{1}{n}\sum_{i=1}^{n} Z_i \to E(Z) \quad \text{(almost surely)}.$$

2. Assume that the random variables are independent and take values in $[0,1]$. Then for any $\varepsilon > 0$,

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} Z_i - E(Z)\right| \geq \varepsilon\right) \leq 2\exp(-2n\varepsilon^2).$$

How close are true and empirical risk of a fixed function f_0? (3)

Direct consequence:

Proposition 2 (Error in measuring the risk for a fixed function)

Fix a function $f_0 \in \mathcal{F}$, and consider a bounded loss function. Draw n training points.

1. Fix some deviation ε, $0 < \varepsilon < 1$. Then,

$$P(|R_n(f_0) - R(f_0)| \geq \varepsilon) \leq 2\exp(-2n\varepsilon^2)$$

2. Equivalently, fix a failure probability δ, $0 < \delta < 1$. Then with probability at least $1 - \delta$,

$$|R(f_0) - R_n(f_0)| \leq \sqrt{\frac{\log(2/\delta)}{2n}}$$

Crucial ingredient: independence!

How close are true and empirical risk of a learned function f_n?

The previous statement was for a function f_0 that we fixed before seeing the training data. Now we would like to get a similar statement for the function f_n that has been learned from n data points:

$$P(|R_n(f_n) - R(f_n)| \geq \varepsilon) \quad \leq \quad ???$$

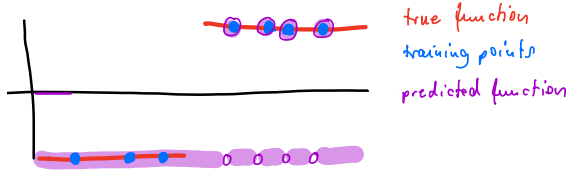
Looks easy, doesn't it? We just apply the previous proposition ... ?!?

But ...let's see.

How close are true and empirical risk of a learned function f_n? (2)

Consider the following example:

- $\mathcal{X} = [0, 1]$ with uniform distribution; labels deterministic with $x < 0.5 \implies y = -1$ and $x \geq 0.5 \implies y = +1$
- Draw n training points
- Define f_n as follows: for all points in the training sample, predict the training label; for all other points predict -1.
- Then for all n we have $R_n(f_n) = 0$ but $R(f_n) = 0.5$.



No convergence!!! Where is the catch?

Analyzing a data-dependent function f_n: uniform convergence

Can we define a condition that is sufficient to establish that the empirical risk of f_n is close to its true risk?

Idea:

- We require that for ALL functions in $\mathcal{F}$, the empirical risk (as measured on the data) has to be close to the true risk.
- If $R_n(f)$ and $R(f)$ are close for ALL functions $f \in \mathcal{F}$, they will in particular be close for the function f_n that has been chosen by the classification algorithm.
- So what we want is that for any $\varepsilon > 0$, with high probability it holds that

$$\sup_{f \in \mathcal{F}} |R_n(f) - R(f)| < \varepsilon$$

Analyzing a data-dependent function f_n: uniform convergence (2)

Definition 3

We say that $R_n(f)$ converges to $R(f)$ **uniformly** over $\mathcal{F}$ if

$$\forall \varepsilon > 0 : \lim_{n \to \infty} P(\sup_{f \in \mathcal{F}} |R_n(f) - R(f)| > \varepsilon) = 0$$

Uniform convergence is sufficient and necessary for consistency

It is elementary to see the following:

Proposition 4 (Uniform convergence is sufficient for consistency)

Let f_n be the function that minimizes the empirical risk in $\mathcal{F}$, and $f_{\mathcal{F}}$ the function that minimizes the true risk in $\mathcal{F}$. Then:

$$P(|R(f_n) - R(f_{\mathcal{F}})| \geq \varepsilon) \quad \leq \quad P(\sup_{f \in \mathcal{F}} |R(f) - \mathrm{R_n}(f)| \geq \varepsilon/2).$$

Main idea of the proof:

$$R(f_n) - R(f_{\mathcal{F}}) = \underbrace{R(f_n) - \mathrm{R_n}(f_n)}_{<\text{orange term}} + \underbrace{\mathrm{R_n}(f_n) - \mathrm{R_n}(f_{\mathcal{F}})}_{<0} + \underbrace{\mathrm{R_n}(f_{\mathcal{F}}) - R(f_{\mathcal{F}})}_{<\text{orange term}}$$

$$\leq 2 \sup_{f \in \mathcal{F}} |R(f) - \mathrm{R_n}(f)|$$

☺

Uniform convergence is sufficient and necessary for consistency (2)

Proposition 5 (Uniform convergence is necessary for consistency)

It is a deep result by Vapnik and Chervonenkis (1971) that uniform convergence is also necessary for (a slightly stricter version) of consistency.

So uniform convergence is what we are after!!!

Uniform convergence: first intuition

How do we know whether we have uniform consistency for some function class $\mathcal{F}$?
The key term that we need to bound is this one:

$$P(\sup_{f \in \mathcal{F}} |R(f) - R_n(f)| > \varepsilon)$$

- If a function class $\mathcal{F}$ is "small" (say, it only consists of a single function), then we have uniform convergence.
- If a function class $\mathcal{F}$ is "too large" (as in the counter-example), then we don't have uniform convergence.

We now want to come up with ways to measure the "size" of a function class.

Generalization bound for finite classes

Let's start with a simple case, a function class with finitely many functions: $\mathcal{F} = \{f_1, ..., f_m\}$.

Proposition 6 (Generalization of finite classes)

Assume $\mathcal{F}$ is finite and contains m functions. Choose any deviation $\varepsilon, 0 < \varepsilon < 1$.

1. Then, with probability at least $1 - 2m\exp(-2n\varepsilon^2)$, we have for all $f \in \mathcal{F}$ that

$$|R(f) - R_n(f)| < \varepsilon.$$

2. Equivalently, choose some failure probability $0 < \delta < 1$. Then, with probability at least $1 - \delta$, for all $f \in \mathcal{F}$ we have

$$R(f) \leq R_n(f) + \sqrt{\frac{\log(2m) + \log(1/\delta)}{2n}}$$

Generalization bound for finite classes (2)

Proof idea:

We exploit (1) the union bound and then (2) our known bound for a single function:

$$
\begin{aligned}
&\mathrm{P}(\sup_{f \in \mathcal{F}} |R(f) - R_n(f)| \geq \varepsilon) \\
&\quad = \mathrm{P}(\sup_{i=1,\ldots,m} |R(f_i) - R_n(f_i)| \geq \varepsilon) \\
&\quad \overset{(1)}{=} \mathrm{P}\left(|R(f_1) - R_n(f_1)| \geq \varepsilon \text{ or } |R(f_2) - R_n(f_2)| \geq \varepsilon \text{ or } \ldots \right) \\
&\quad \overset{(2)}{\leq} \sum_{i=1}^{m} \mathrm{P}(|R(f_i) - R_n(f_i)| \geq \varepsilon) \\
&\quad \leq 2m \exp(-2n\varepsilon^2)
\end{aligned}
$$

☺

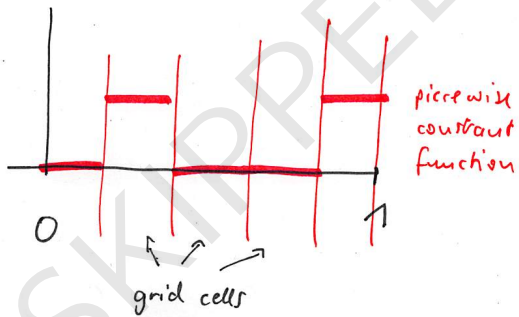
Generalization bound for finite classes (3)

Let's digest this bound:

- It bounds the true risk by the empirical risk plus a "capacity term".
- If the function class gets larger (m increases), then the bound gets worse.
- If m is "small enough" compared to n (in the sense that $\log m/n$ is small), then we get a tight bound.
- The sample complexity of learning from a finite class scales as $\sqrt{\log m/n}$. For the size of the function class we pay logarithmically in m.
- If you want to grow your function space with n in order to be able to fit more accurately if you have more data, you need to make sure that $\log m/n \to 0$ if you want to get consistency.

Example: Partitioning-based estimators

Consider $\mathcal{X} = [0, 1]$, split it into a grid of k cells of the same size. As function class, consider all functions that are piecewise constant (0 or 1) on all cells. Denote this function class by $\mathcal{F}_k$.



SKIPPED

We want to study the question how to choose the number k of partitions depending on the sample size n.

Example: Partitioning-based estimators (2)

Intuition:

- To get the variance to 0, we need to see "infinitely many points" in each of the intervals, to consistently estimate the mean.
- To get the bias to 0, we need the number of partitions k to go to ∞, because we need to approximate a possibly complicated Bayes function with a piecewise constant function.

SKIPPED

Partitioning-based estimators in 1d, k fixed

Case 1: k is fixed.

- Then ERM is uniformly consistent with respect to $\mathcal{F}_k$.

  Proof: The function class has 2^k many functions. Now just apply the above bound: The key error term is of the order $\log(m)/n = \log(2^k)/n \approx k/n$, and this goes to 0 if $n \to \infty$ and k is fixed.

SKIPPED

Partitioning-based estimators in 1d, k growing with n

Case 2: k grows with n, so formally k is a function of n, denoted by $k(n)$.

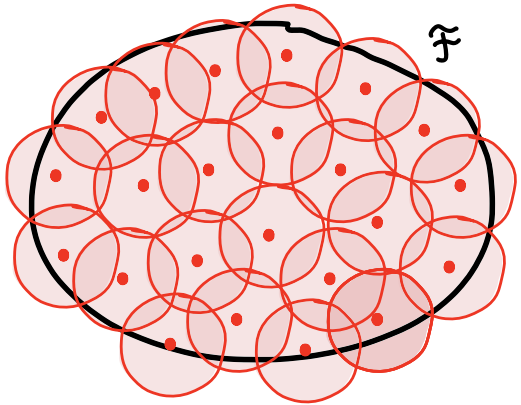
- How fast can $k(n)$ grow such that we still have consistency with respect to $\mathcal{F}_{k(n)}$?
- By the same argument as above, the error term is of the order $k(n)/n$. So the bound only goes to 0 if $k(n)/n \to 0$. For example, we could choose $k(n) = \log n$.
- What about the approximation error / Bayes consistency?

  If $k(n)/n \to 0$, the estimation error goes to 0.
  If we also want that the approximation error goes to 0 we additionally need to require $k \to \infty$. One can now invoke maths / measure theory to prove that $k \to \infty$ really allows us to approximate the Bayes classifier by finite histograms, for all probability measures P (skipped here).
  So if we have both $k \to \infty$ and $k(n)/n \to 0$, and if our distribution satisfies some regularity assumptions (so that it can be well-approximated by finite intervals), then we will get Bayes consistency.

SKIPPED

How to deal with infinite function classes

Approach 1: geometric argument with covering numbers



Cover $\mathcal{F}$ with m small balls (finitely many!)

Control deviations within a ball

Overall error = finite class error with m functions + deviations

Approach 1: geometric argument with covering numbers (2)

Proposition 7 (Simple bound with covering numbers)

Fix an error probability $0 < \delta < 1$. Fix a radius $r > 0$ and a norm $\|\cdot\|$ on $\mathcal{F}$. Assume

- that $\mathcal{F}$ can be covered by $C(\mathcal{F}, \|\cdot\|, r)$ many $\|\cdot\|$-balls of radius r
- that there exists $\eta > 0$ such that for all centers $f \in \mathcal{F}$ and for all $f_1, f_2 \in B_r(f)$, $R(f_1) - R(f_2) \leq \eta$ and $R_n(f_1) - R_n(f_2) \leq \eta$

Then with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,

$$R(f) \leq R_n(f) + \sqrt{\frac{\log(2C(\mathcal{F}, \|\cdot\|, r)) + \log(1/\delta)}{2n}} + 2\eta$$

Approach 1: geometric argument with covering numbers (3)

Proof idea: Assume that $\mathcal{F}$ is covered by m balls with centers $f_1, ..., f_m \in \mathcal{F}$.

- For an arbitrary $f \in \mathcal{F}$, let f_i the closest center of a ball to f. Then:

$$\begin{aligned} |R(f) - R_n(f)| &= |R(f) - R(f_i) + R(f_i) - R_n(f_i) + R_n(f_i) - R_n(f)| \\ &\leq |R(f) - R(f_i)| + |R(f_i) - R_n(f_i)| + |R_n(f_i) - R_n(f)| \end{aligned}$$

- Then bound overall worst case error over $\mathcal{F}$:

$$\sup_{f \in \mathcal{F}} |R_n(f) - R(f)| \leq \sup_{f \in \{f_1, ..., f_m\}} |R_n(f) - R(f)| + 2\eta$$

- To the red term, apply our previous bound for a finite function class.

☺

Example: linear regression with absolute loss

- Consider regression on $[0,1]$ with linear functions with bounded slope: $\mathcal{F} = \{h_a(x) = ax : a \in [0,1]\}$.
- For an unknown target function $g : [0,1] \to [0,1]$, we consider the L_1-loss function: $\ell(h(x), g(x)) = |h(x) - g(x)|$
- We consider η-balls in $\mathcal{F}$ with respect to the infinity-norm:

$$B_\eta(h) = \{\tilde{h} \in \mathcal{F} : \|h - \tilde{h}\|_\infty \leq \eta\}.$$

Observe that for two functions with slopes a and b we have

$$\|h_a - h_b\|_\infty = \sup_{x \in [0,1]} |ax - bx| = |a - b|$$

So if $h_a \in B_\eta(h_b)$ then $|a - b| \leq \eta$.

SKIPPED

Example: linear regression with absolute loss (2)

- For any $h_a \in B_\eta(h_b)$, we can bound their difference in loss:
  For every x,

$$\left|\ell(h_a(x), g(x)) - \ell(h_b(x), g(x))\right| \leq |h_a(x) - h_b(x)| \leq \eta.$$

- To cover the space, we can use the balls with slopes $a_0 = 0, a_1 = \eta, a_2 = 2\eta, \ldots$ (observe that by definition, the maximal slope of our functions is 1). Overall, we need of the order $1/\eta$ many balls to cover the space, so $C(\mathcal{F}, \eta) \approx 1/\eta$.
- Altogether, with probability at least $1 - \delta$, we have that for all $f \in \mathcal{F}$,

$$R(f) \leq R_n(f) + 2\eta + \sqrt{\frac{\log(2/\eta) + \log(1/\delta)}{2n}}.$$

- Note that we can choose any η we want. For example, if we choose $\eta = 1/n$ we obtain an overall bound of the order $\sqrt{log(n)/n}$.

SKIPPED

Example: linear regression with absolute loss (3)

Extending this example to a d-dimensional space, one will find that the covering numbers are of the order of $C(\mathcal{F}, \eta) = (1/\eta)^d$. So the crucial term in the bound gets an extra factor of d:

$$\log(C(\mathcal{F}, \eta)) = d \log(1/\eta)$$

SKIPPED

Approach 2: Combinatorial arguments with shattering coefficient

In case of binary classification: even if we have infinitely many functions in $\mathcal{F}$, on the training data they can only "do finitely many things".

Definition: For a given sample $X_1, ..., X_n \in \mathcal{X}$ and a function class $\mathcal{F}$ define $\mathcal{F}_{X_1,...,X_n}$ as the set of those functions that we get by restricting $\mathcal{F}$ to the sample:

$$\mathcal{F}_{X_1,...,X_n} := \{f|_{X_1,...,X_n}; f \in \mathcal{F}\}$$

The shattering coefficient $\mathcal{N}(\mathcal{F}, n)$ of a function class $\mathcal{F}$ is defined as the maximal number of functions in $\mathcal{F}_{X_1,...,X_n}$:

$$\mathcal{N}(\mathcal{F}, n) := \max\{|\mathcal{F}_{X_1,...,X_n}| \; ; \; X_1, ..., X_n \in \mathcal{X}\}$$

Shattering coefficient: generalization bound

Theorem 8 (Generalization bound with shattering coefficient)

Let $\mathcal{F}$ be any arbitrary function class. Then for all $0 < \varepsilon < 1$,

$$\mathrm{P}(\sup_{f \in \mathcal{F}} |R(f) - R_n(f)| > \varepsilon) \leq 2\mathcal{N}(\mathcal{F}, 2n) \exp(-n\varepsilon^2/4).$$

The other way round: With probability at least $1 - \delta$, all functions $f \in \mathcal{F}$ satisfy

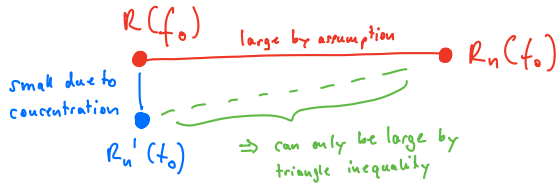
$$R(f) \leq R_n(f) + 2\sqrt{\frac{\log(2\mathcal{N}(\mathcal{F}, 2n)) - \log(\delta)}{n}}.$$

Key ingredient in the proof: symmetrization lemma

- By R_n we denote the risk on our given sample of n points.
- By R'_n we denote the risk that we get on a second, independent sample of n points, called the "ghost sample".

Proposition 9 (Symmetrization lemma)

$$\mathrm{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) \;\leq\; 2\,\mathrm{P}(\sup_{f\in\mathcal{F}} |R_n(f) - R'_n(f)| > \varepsilon/2).$$



Proof intuition:

Key ingredient in the proof: symmetrization lemma (2)

- Draw the original sample.
- If $\sup_{f \in \mathcal{F}} |R(f) - R_n(f)|$ is large, this can only be the case if there exists some $f_0 \in \mathcal{F}$ such that $|R(f_0) - R_n(f_0)|$ is large. Fix this function f_0.
- Now draw the independent ghost sample. Because f_0 has been fixed in advance, with high probability with have concentration: $|R(f_0) - R'_n(f_0)|$ is small.
- But together, this implies that $|R'_n(f_0) - R_n(f_0)|$ is large (see figure).
- If it is already large for f_0, the sup can only make it larger.

Proof of the generalization bound

Sketch:

- First apply the symmetrization lemma.
- Its right hand side only depends on the values of the functions f on the two samples: If two functions f and g coincide on all points of the orignal sample and the ghost sample, that is $f(x) = g(x)$ for all x in the samples, then $R_n(f) = R_n(g)$ and $R'_n(f) = R'_n(g)$.
- So the supremum over $f \in \mathcal{F}$ in fact only runs over finitely many functions: all possible binary functions on the two samples.
- The number of such functions is bounded by the shattering coefficient $\mathcal{N}(\mathcal{F}, 2n)$ (we have $2n$ in the shattering coefficient because of n points both in original and in ghost sample)
- Now the shattering bound follows from the bound for finite function classes.

☺

Discussion of the shattering bound

- The shattering coefficient measures "how powerful" a function class is: how many different labelings of a data set it can possibly realize.
- To get consistency of a classifier, we need to establish that $\log \mathcal{N}(\mathcal{F}, 2n)/n \to 0$ as $n \to \infty$.

  Intuitively: the number of possibilities in which a data set can be labeled has to grow at most polynomially in n.

Much more to discover

The shattering coefficient is just one particular "capacity measure" of function classes, there are many more out there. The two most popular ones:

- VC dimension: a combinatorial coefficient that allows to control the shattering coefficient of function classes. But often difficult to determine.
- The Rademacher complexity of a function class $\mathcal{F}$ with respect to n is defined as

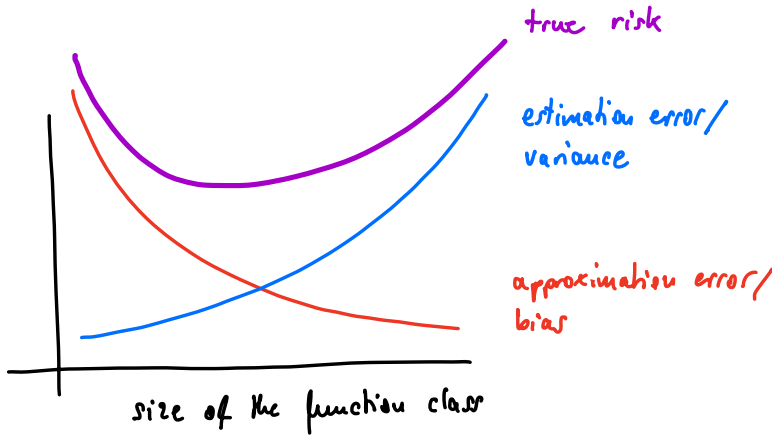
$$\mathrm{Rad}_n(\mathcal{F}) := E \sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(X_i)$$

  where σ_i are random labels.

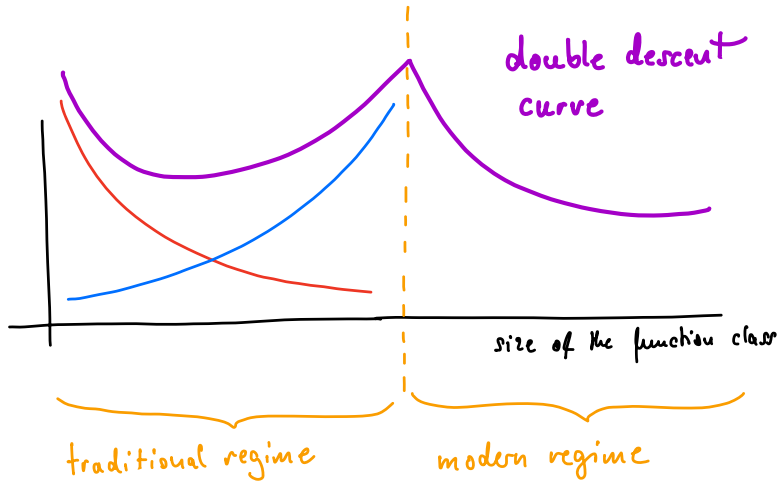
If I would like to prove a generalization bound for a particular function class, I would always attempt the Rademacher complexity first (easiest to work with).

.

Step back: what have we achieved?



Outlook to next lectures



Lecture 2: Double descent phenomenon and minimum norm solution

Text books:

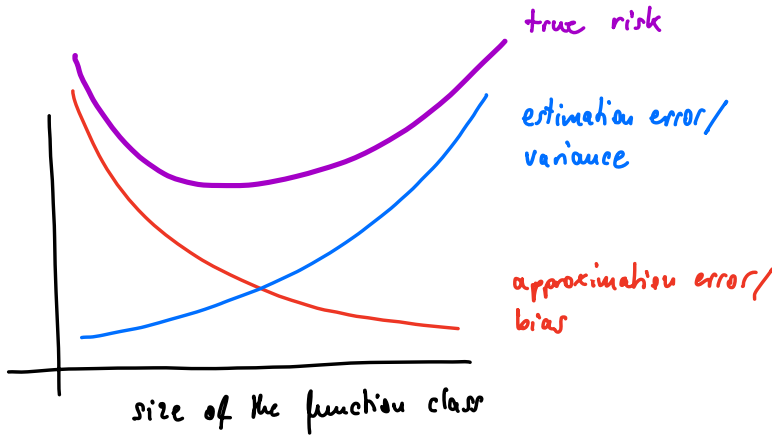
- Francis Bach: Learning theory from first principles. 2025
- Hardt/Recht: Patterns, predictions, actions. 2022

The double descent curve

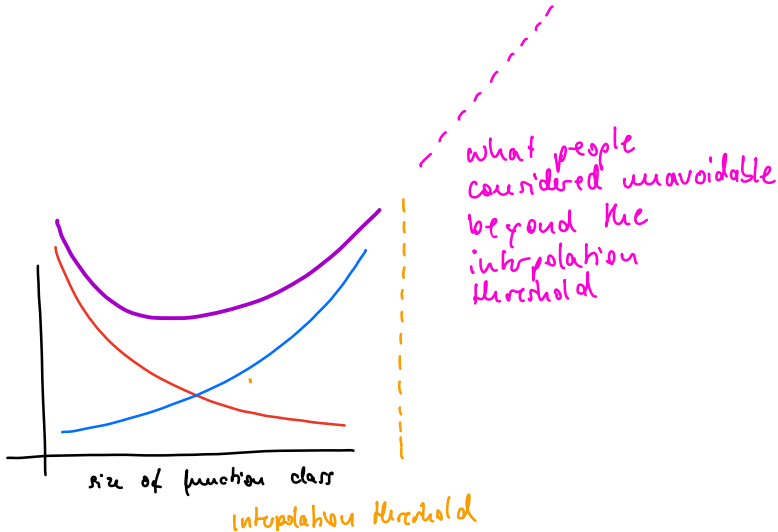
Original papers:

- Mikhail Belkin, Daniel Hsu, Siyuan Ma, Soumik Mandal: Reconciling modern machine-learning practice and the classical bias-variance trade-off. PNAS, 2019.
- Mikhail Belkin: Fit without fear, remarkable mathematical phenomena of deep learning through the prism of interpolation. Acta Numerica, 2021.
- Peter Bartlett, Andrea Montanari, Sasha Rakhlin: Acta Numerica, 2021.

Last lecture

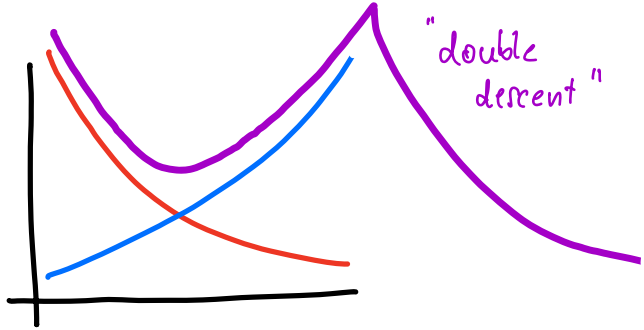


Last lecture (2)



But then ...

... deep neural networks became successful. They tend to work in the overparameterized regime (more parameters than data points) and often are trained to complete overfitting (training error 0) and beyond. Yet, generalization can take place ...



Double descent curve

Radically new insight: Need to distinguish:

Under-parameterized regime:

- fewer parameters than data points: $n > d$
- cannot interpolate
- "traditional" behavior

Over-parameterized / interpolation regime:

- more parameters than data points: $n < d$
- interpolation possible, often training error 0 can be achieved
- Often there even exist many interpolating functions. Many of them do not generalize — but some of them possibly do.

Double descent curve (2)

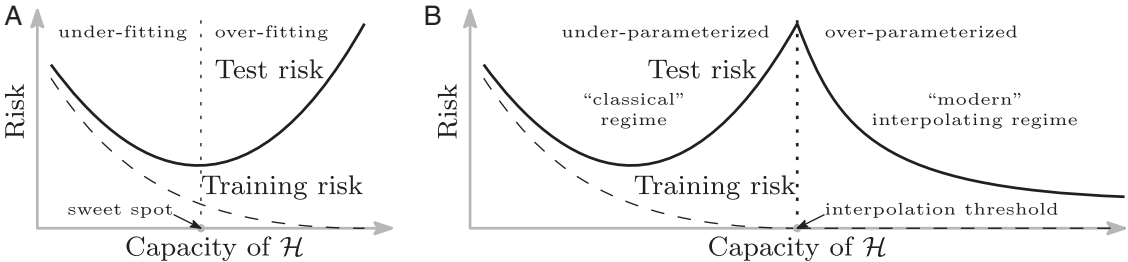


Fig. 1. Curves for training risk (dashed line) and test risk (solid line). (*A*) The classical U-shaped risk curve arising from the bias–variance trade-off. (*B*) The double-descent risk curve, which incorporates the U-shaped risk curve (i.e., the "classical" regime) together with the observed behavior 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.

Figure from Belkin et al, 2019, PNAS

In their original paper, such behavior has been established empirically for a range of algorithms:

Double descent curve (3)

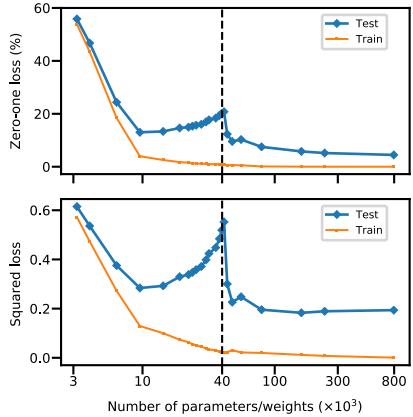


Figure 4: **Double descent risk curve for fully connected neural network on MNIST.** Training and test risks of network with a single layer of H hidden units, learned on a subset of MNIST ($n = 4 \cdot 10^3$, $d = 784$, $K = 10$ classes). The number of parameters is $(d+1) \cdot H + (H+1) \cdot K$. The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

Double descent curve (4)

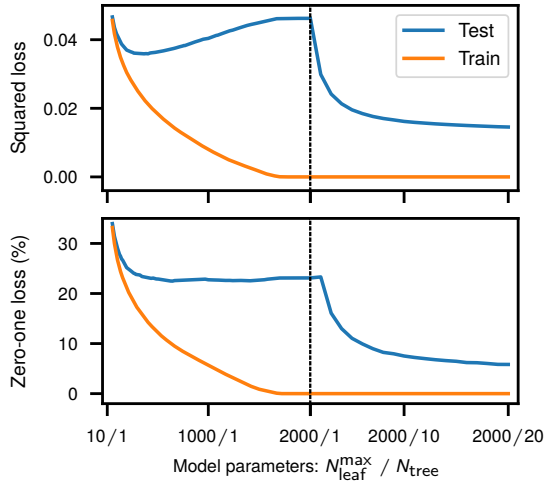
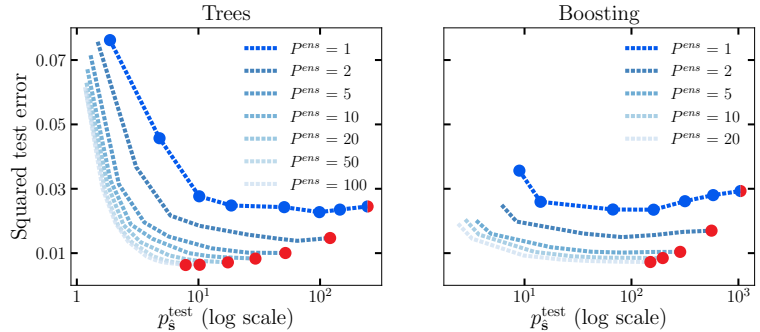


Figure 5: **Double descent risk curve for random forests on MNIST.** The double descent risk curve is observed for random forests with increasing model complexity trained on a subset of MNIST ($n = 10^4, 10$ classes). Its complexity is controlled by the number of trees N_{tree} and the maximum number of leaves allowed for each tree $N_{\text{leaf}}^{\text{max}}$.

Other descent curves

It is not the case that the training/test error curves always show a double descent, many more shapes are possible:

- A single descent:



Curth et al, U-turn on double descent, NeurIPS 2023

Other descent curves (2)

- Triple descent:

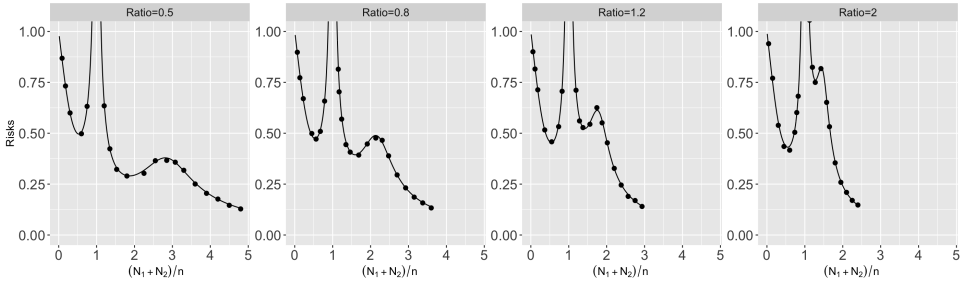
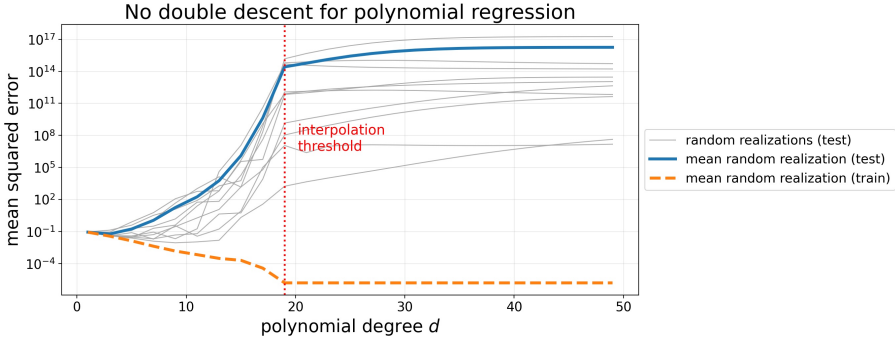


Figure 6: Risk curves of DRFMs with different ratios between random feature dimensions. The plots show both the asymptotic excess risks (curves) and empirical excess risks (dots). From (a) to (d), the ratios N_1/N_2 are 0.5, 0.8, 1.2 and 2, respectively. The activation functions are chosen as $\sigma_1(x) = \mathrm{ELU}(3x)$ and $\sigma_2(x) = \mathrm{ReLU}(x/4)$.

Meng et al: Multiple descent in the multiple random feature model, JMLR 2024

A case where double descent does NOT happen

Just to make sure this is clear: double descent does not automatically happen!!! Here is an example where overfitting beyond the interpolation threshold goes wrong:



The setup:

A case where double descent does NOT happen (2)

The setup:

- We consider $n = 20$ training points x_i sampled uniformly from $[-1, 1]$, with noisy labels

$$y_i = \frac{1}{1 + 25x_i^2} + \varepsilon_i.$$

- We fit polynomial regression models of increasing degree d, using the monomial features $1, x, x^2, \ldots, x^d$ with least squares loss:

$$\min_{a \in \mathbb{R}^{d+1}} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{k=0}^{d} a_k x_i^k \right)^2.$$

  The interpolation threshold occurs at $d = 19$.

- The test error is evaluated on a dense grid in $[-1, 1]$. Beyond the interpolation threshold, the high-degree polynomial becomes unstable and the test error grows, rather than exhibiting a second descent.

Learning in the overparameterized regime

The key take-home message: learning in the over-parameterized regime can work successfully! The exact shape of the curve is not important. What is important is that the curve might (but not always does) exhibit smaller test error beyond the interpolation threshold.

But how can we theoretically understand what is going on???

Example of linear least squares in the overparameterized regime

Setup:

- Training data: $X_1, \ldots, X_n \in \mathbb{R}^d$, labels $y_1, \ldots, y_n \in \mathbb{R}$. For concise notation, stack the points in a matrix $X \in \mathbb{R}^{n \times d}$, the labels in a vector $y \in \mathbb{R}^n$.
- We now try to fit a linear model with parameters $\theta \in \mathbb{R}^d$ to the training data: $f_\theta(x) = x^T \theta$.

Overparameterized regime:

- We consider the overparameterized regime $d > n$.
- For simplicity we assume that the data matrix X has rank n (this is often the case, for example if we sample training points from a Gaussian).
- Observe: X is a linear mapping from $\mathbb{R}^d \to \mathbb{R}^n$. In the overparameterized regime $d > n$, there exist many vectors v with $Xv = 0$.

Example of linear least squares in the overparameterized regime (2)

Many exact solutions in the overparameterized regime:

- In the overparameterized regime, there always exists a solution θ^* that perfectly fits the training data, that is it satisfies $X\theta^* = y$. For example, one such solution is given by $\theta^* = X^T(XX^T)^{-1}y$.
- But there exist many more solutions: for any vector v with $Xv = 0$, the vector $\tilde{\theta} = \theta + v$ is also a solution!
- And many of these solutions would behave very differently on test data (for example, in directions that we did not observe in the training data).

The puzzle:

- All these solutions have zero training error, but the test error can be very high!
- So why on earth would we see double descent??? (Would we?)

Implicit regularization by the choice of the optimization algorithm

GD and SGD for linear least squares regression converge to the minimum norm solution

Text book: training data consists of $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$
Bach book, Sec 12.1.1. (p. 344)

Setup: GD, linear least squares, over-parameterized

We consider linear least squares regression in the over-parameterized regime:

- Training data $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, $d > n$
- We assume X has rank n, so perfect interpolation is possible (as we just discussed).
- As learning algorithm we consider gradient descent with the least-squares objective $L(w) = \frac{1}{2}\|Xw - y\|^2$:
  - initialized at $w_0 = 0$
  - update steps
    $$w_{t+1} = w_t - \eta X^\top (Xw_t - y).$$
  - constant step size satisfying $0 < \eta < 2/\lambda_{\max}(XX^\top)$ where $\lambda_{\max}$ denotes the largest eigenvalue of the matrix.

Convergence to minimum norm solution

Theorem 10 (Convergence to minimum norm solution)

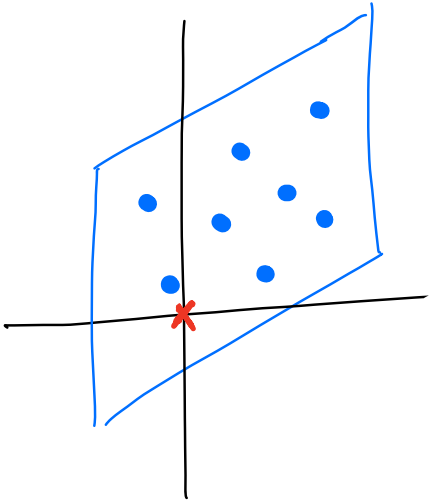
In the setup of the previous page, the following statements are true:

(1) The GD iterates w_t converge to $w^\star = X^\top (XX^\top)^{-1} y$.

(2) $w^\star$ is the minimum-norm solution: among all the solutions of $Xw = y$ it is the one with minimal Euclidean norm $\|w\|$. It is the unique solution with this property.

A similar result also holds for SGD.

Proof sketch

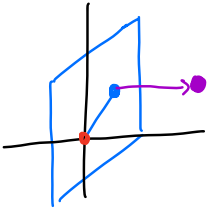


data lives in n-dim subspace S_n

we start in 0, no "incentive" to ever move out

Inside S_n there only exists one unique solution: $X^T(XX^T)^{-1}y$

All other solutions are the sum of this solution plus an orthogonal term. Their norm is then always $\geq$ the one of the other solution



Proof

Step 1: All iterates w_t lie in $S_n := span\{x_1, ..., x_n\} = \text{range}(X^\top)$: Proof by induction.

- Obviously true for $w_0 = 0$.
- To treat the general case $t > 0$, observe that the update term $\eta X^\top (X w_t - y)$ is a linear combination of training points (because no matter what v is, it is true that $X^\top v = \sum_{i=1}^n v_i x_i$). Thus, if $w_t \in S_n$, also $w_{t+1} \in S_n$, so we can express them in the form $w_t = X^T \alpha_t$ for some $\alpha_t \in \mathbb{R}^n$.

SKIPPED

Proof (2)

Step 2: The residuals satisfy $r_{t+1} = (I - \eta XX^\top)\, r_t$.
Define the residual $r_t := Xw_t - y$. Then we have:

$$
\begin{aligned}
r_{t+1} &= Xw_{t+1} - y \text{ (plug in GD update rule)} \\
&= X(w_t - \eta X^\top (Xw_t - y)) - y \\
&= Xw_t - \eta XX^\top (Xw_t - y) - y \\
&= (Xw_t - y) - \eta XX^\top (Xw_t - y) \text{ (use def of } r_t\text{)} \\
&= r_t - \eta XX^\top r_t \\
&= (I - \eta XX^\top) r_t
\end{aligned}
$$

SKIPPED

Proof (3)

Step 3: Gradient descent is contractive and converges to an interpolating solution. Since $XX^\top$ is symmetric positive semi-definite and $0 < \eta < 2/\lambda_{\max}(XX^\top)$, all eigenvalues of $I - \eta XX^\top$ have modulus strictly smaller than 1. Hence $I - \eta XX^\top$ is a strict contraction. Consequently, $r_t \to 0$ and hence $Xw_t \to y$.

Step 4: Solution is given by $w^\star = X^\top(XX^\top)^{-1}y$:

- We already know that $Xw_t \to y$ (by Step 3) and $Xw_t = XX^\top\alpha_t$ for some α_t (by Step 1), and $XX^\top$ is invertible (by assumption).
- Thus $\alpha_t \to (XX^\top)^{-1}y$, and thus $w_t = X^\top\alpha_t \to X^\top(XX^\top)^{-1}y$.

SKIPPED

Proof (4)

Step 5: Inside S_n there only exists one unique solution. S_n is an n-dim subspace, and by assumption rank X is n, so the linear system can have at most one solution in S_n:

if $w = X^\top \alpha \in S_n$ and $Xw = y$, then $XX^\top \alpha = y$. Since $XX^\top$ is invertible, $\alpha = (XX^\top)^{-1} y$ is unique.

Step 6: Minimum norm solution.
We claim that among all interpolating solutions, the unique one in range$(X^\top)$ is the minimum-norm solution.

- Let w be any solution: $Xw = y$.
- Decompose it orthogonally as $w = w_\parallel + w_\perp$, where $w_\parallel \in \text{range}(X^\top)$ and $w_\perp \in \ker(X)$.
- Observe that $Xw = Xw_\parallel$ (because by definition $w_\perp \in ker(X)$, thus $Xw_\perp = 0$), so $w_\parallel \in S_n$ is also a solution to our problem.

SKIPPED

Proof (5)

- In particular, because we already know by Step 5 that the solution in S_n is unique, the part $w_{\|}$ is exactly the same for all solutions w, namely $w_{\|} = w^\star$.
- Finally, $\|w\|^2 = \|w_{\|}\|^2 + \|w_\perp\|^2$. Hence the norm is minimized iff $w_\perp = 0$, i.e. iff $w = w_{\|} = w^\star$.

☺

SKIPPED

Discussion

The result shows that GD performs "implicit regularization": among the many solutions that interpolate the data, it always selects the one that has the smallest norm.

In many other scenarios, we would try to achieve the same by explicit regularization. Here it happens automatically, implicitly.

In particular, the choice of the optimization algorithm (GD or SGD) ensures that we not only find a solution that has a small objective value, but also that has other nice properties.

This is not obvious at all, and other procedures for solving the optimization problem might not have this property! (and thus, potentially, might not generalize).

GD for logistic loss classification converges to maximum margin solution

Literature: Bach book, Sec. 12.1.2.

Convergence to max margin solution

Theorem 11 (Implicit bias of GD for logistic regression)

Consider data $(x_i, y_i)_{i=1,\dots,n}$ for binary classification, that is $x_i \in \mathbb{R}^d$, $y_i \in \{\pm 1\}$. Assume the data is linearly separable (so we can "interpolate"). We consider the logistic loss function $L(w) = \sum_{i=1}^n \log\left(1 + e^{-y_i \langle w, x_i \rangle}\right)$ and run gradient descent from any initialization with sufficiently small, constant step size. Then the GD iterates diverge ($\|w_t\| \to \infty$), but the direction converges:

$$\frac{w_t}{\|w_t\|} \to \frac{w_{\mathrm{mm}}}{\|w_{\mathrm{mm}}\|},$$

where w_{mm} is the maximum-margin linear separator:

$$w_{\mathrm{mm}} = \arg\min_w \|w\|^2 \quad \text{s.t. } \forall i: \; y_i \langle w, x_i \rangle \geq 1$$

Reference: Soudry, Hoffer, Nacson, Gunasekar, Srebro: "The Implicit Bias of Gradient Descent on Separable Data," JMLR 2018.

Convergence to max margin solution (2)

Proof: skipped, see the Bach book if you are interested.

Note that a related result for SGD exists, but one needs stronger assumptions.

Implicit bias of neural networks

Implict bias on neural networks

There also exist some results describing the implicit bias of neural networks, for example:

- Wide, shallow networks converge to the minimum RKHS norm interpolant in the corresponding kernel space (Jacot et al., 2018; Arora et al., 2019).
- Deep linear networks trained with gradient descent converge towards maximum-margin predictors with respect to an architecture-dependent norm (Gunasekar et al., 2018).
- Homogeneous neural networks trained on separable data often maximize margin under an implicit path-type norm (Lyu,Li, 2019).
- In matrix factorization and low-rank models, gradient descent implicitly favors low-rank / minimum nuclear norm solutions (Gunasekar et al., 2017).

Implict bias on neural networks (2)

However:

- For nonlinear deep networks trained with SGD, the implicit bias is not very well understood and may even depend on optimization noise and initialization (Blanc et al., 2020; Damian et al., 2021).
- So to a certain degree, the idea of implicit regularization can formally explain many simple cases, but no strong results (yet?) exist for complex architectures as they are often used in practice.

Lecture 3: Benign overfitting

Literature:

Not really treated in text books. Here are some of the key original papers. Many of them present intuition as well, but the maths is difficult:

- Benign overfitting in linear regression Peter Bartlett, Philip M Long Gabor Lugosi, Alexander Tsigler, PNAS 2020
- Tsigler, A., Bartlett, P. L. (2023). Benign overfitting in ridge regression. Journal of Machine Learning Research, 24(123), 1-76.
- Hastie, Montanari, Rosset, Tibshirani: Surprises in high-dimensional ridgeless least squares interpolation. Annals of Statistics, 2022
- Deep learning: a statistical viewpoint (Peter Bartlett, Andrea Montanari, Sasha Rakhlin, 2021).

What is benign overfitting?

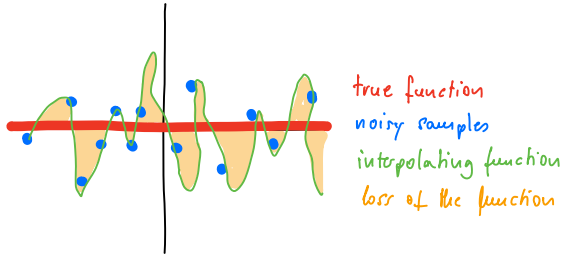
- Classical statistical wisdom would suggest: interpolating many data points leads to bad overfitting: we have a low training error but a high test error.
- However, in modern high-dimensional models, this intuition can fail: in DNNs we often fit to training error 0, yet still have a good test performance.
- Situations in which this happens have been called benign overfitting. This term was coined in the paper by Bartlett, Long, Lugosi, Tsigler: "Benign overfitting in linear regression."
- How can a function interpolate noise and still predict well?
- We will study this question for different setups of linear regression.

First toy setup: Learning the constant 0 function

Toy setup: isotropic Gaussian input, constant 0 output with noise

We first consider a toy setup where the function that we want to learn is constant 0, and all we observe are noisy observations of this function.

"Classical intuition" would lead us to think that if we interpolate the noise, it cannot generalize, and the test error would be large.



The trick is now: while this intuition might be correct for many interpolating solutions, it is wrong if we use a very specific solution: the minimum norm interpolator.

Toy setup: isotropic Gaussian input, constant 0 output with noise (2)

Formal setup:

- We consider a regression problem with input points $x_1, .., x_n \in \mathbb{R}^d$, drawn iid from $N(0, I_d)$. Denote by $X \in \mathbb{R}^{n \times d}$ the data matrix.
- We assume the true output is equal to 0 everywhere (the Bayes predictor is the constant 0 function).
- We assume that we observe the true label 0 with iid additive noise, that is $y_i \sim N(0, \sigma^2)$.
- We consider linear prediction in the over-parameterized regime where $n + 1 < d$, with squared loss.

Benign overfitting in toy setup

Theorem 12 (Benign overfitting in simple setup)

Consider the minimum-norm interpolator $\hat{w} = X^\top (XX^\top)^{-1} y$ for the setup of the previous slide. Its expected true risk with respect to the least squares loss function is given by

$$\mathbb{E}_{X,y,x}\big((x^\top \hat{w} - 0)^2\big) = \sigma^2 \frac{n}{d-n-1}.$$

Here $E_{X,y,x}$ denotes the expectation with respect to the training points X, their labels y and the test point x.

In particular, if the dimension d is very large compared to n, then the excess prediction risk tends to 0, although $\hat{w}$ interpolates pure noise!!!.

Proof sketch

Proof sketch.

- The matrix X has size $n \times d$, with $d > n$. One can prove that for Gaussian random variables, the $n \times n$-matrix $XX^\top$ is invertible almost surely. Therefore the least squares solution $\hat{w} = X^\top (XX^\top)^{-1} y$ perfectly interpolates the training data:

$$X\hat{w} = XX^\top (XX^\top)^{-1} y = y.$$

- For a test point $x \sim N(0, I_d)$, the true expected squared error of the prediction $\hat{y} = x^t \hat{w}$ is given as

$$E_x((\hat{y} - 0)^2) = E_x((x^t \hat{w})^2) = E_x(\hat{w}^t x x^t \hat{w}) = \hat{w}^t E_x(xx^t)\hat{w} = \hat{w}^t I_d \hat{w} = \|\hat{w}\|^2$$

- We can compute this norm by plugging in the expression of $\hat{w}$:

$$\|\hat{w}\|^2 = \langle \hat{w}, \hat{w} \rangle = y^\top (XX^\top)^{-1} XX^\top (XX^\top)^{-1} y = y^\top (XX^\top)^{-1} y.$$

SKIPPED

Proof sketch (2)

- We now want to compute the expected value of this quantity: $E_{X,y}\|\hat{w}\|^2$. We do it in two steps: first we treat X as fixed and compute the expectation wrt y conditioned on the fixed X, and then we compute the expectation wrt X.
  On the way we exploit two facts:
  - For any fixed matrix A and the random vector y we have $E_y(y^\top Ay) = \mathrm{tr}(A\ E_y(yy^\top))$:

$$
\begin{aligned}
E_y(y^\top Ay) &= (\text{because for a real number } a, \text{ we have } a = \mathrm{tr}(a)) \\
&= E_y(\mathrm{tr}(y^\top Ay)) \\
&= E_y(\mathrm{tr}(Ayy^\top)) \\
&= (\text{linearity of } E \text{ and } tr) \\
&= \mathrm{tr}(E_y(Ayy^\top)) \\
&= \mathrm{tr}(A\ E_y(yy^\top))
\end{aligned}
$$

SKIPPED

Proof sketch (3)

So in our case,

$$E_y(y^\top (XX^\top)^{-1} y) = \mathrm{tr}\left((XX^\top)^{-1} E(yy^\top) \right)$$

- $E_y(yy^\top) = \sigma^2 I_n$ because y are the Gaussian noise variables

- Now we need to take the expectation with respect to X. We exploit that $XX^\top$ is a Wishart matrix with d degrees of freedom, and for such a matrix and $d > n+1$ one can prove that $E((XX^\top)^{-1}) = I_n/(d-n-1)$, where I_n denotes the identity matrix.
- Taken together, one can conclude that

$$E_{X,y}\|\hat{w}\|^2 = \mathrm{tr}\left(\frac{1}{d-n-1} I_n \sigma^2 I_n \right) = \sigma^2 \frac{1}{d-n-1} \mathrm{tr}(I_n) = \sigma^2 \frac{n}{d-n-1}.$$

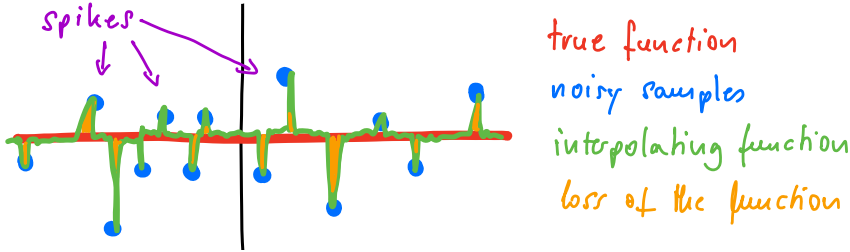
☺

SKIPPED

Original sources: Hastie, Montanari, Rosset, Tibshirani (2019); Bartlett, Long, Lugosi, Tsigler (2020).

What happens geometrically? Spiky-smooth structure of the solution

While the theorem is mathematically correct, it is still difficult to grasp what happens. Here is an illustration:



Large "spikes" interpolate the noisy labels y_i; in high dimensions, these regions have vanishing volume. Outside the spikes, the function approximates the true function (the 0-function in this case).

What happens geometrically? Spiky-smooth structure of the solution (2)

What is the reason for this behavior? Intuition in 5 steps:

Step 1: Prediction function in span of data

We have already seen above that the solution vector lies in the span of our data points, and thus our prediction function has the form $f(x) = \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$

Step 2: Prediction function interpolates $\rightsquigarrow$ tips of the spikes

By assumption, the function interpolates the training points. Here it takes potentially large values. These are the "tips of the spikes"

What happens geometrically? Spiky-smooth structure of the solution (3)

Step 3: The significant part of the spike We look at the region around a training point where the function still has large value. We call this a "spike":

Formally, we define the "spike" $S_i(t)$ of training point x_i with parameter t as the set

$$S_i(t) = \{x \in \mathbb{R}^d \Big| \langle \frac{x}{\|x\|}, \frac{x_i}{\|x_i\|} \rangle > t\}$$



Observe that the red term describes a projection of the vector x on the vector x_i (or vice versa).

What happens geometrically? Spiky-smooth structure of the solution (4)

Step 4: How likely does a random test point sit inside the spikes?
In other words: what is the likelihood that the projection of one random point on another random point is large?

In high-dim spaces, such random projections are highly concentrated: when we draw the two points randomly, the likelihood that the random projection takes a large value (larger than t) is very small. Let's look at this:

Consider u, v drawn from the unit sphere in $\mathbb{R}^d$ according to the uniform distribution. Then one can prove the following concentration inequality about their scalar product:

$$P(\langle u, v \rangle > t) \leq 2\exp(-cdt^2)$$

where c is a constant, d is the dimension.

What happens geometrically? Spiky-smooth structure of the solution (5)

Step 5: Outside the spikes the function approximates the 0 function.

- In high dimensions, the data points are nearly orthogonal. The learned function must take large values at the training points in order to interpolate. This is possible because each data point essentially "opens up" a new dimension in our data space, and we have enough parameters to memorize the point in this dimension.
- As we have seen above, a random test point, with high probability, is not aligned with any of the training points, so it is outside of the spikes.
- **Now we exploit that we consider the minimum norm solution**: The overall function has norm as small as possible. In particular, away from the training points (in the directions where we have no training points), the solution will have small norm, so the prediction is close to zero. Again this is possible because in high dimensions, there are enough parameters to make the function behave "close to zero" in directions that are nearly orthogonal to the training points.

Not so benign: Linear model with isotropic Gaussian inputs

Literature:

Bach book, Sec.12.2.3;
Bartlett, Long, Lugosi, Tsigler: Benign overfitting in linear regression". PNAS, 2020.
Bartlett, Montanari, Rakhlin: Deep learning, a statistical viewpoint. Acta Numerica, 2021.

Setup: linear model, isotropic Gaussian sampling

We are now going to investigate whether similar results as above hold in a slightly more general setting: linear regression with Gaussian input variables. We will see: partially yes, but there are some caveats!

- Input distribution: $x_1, \ldots, x_n \sim N(0, I_d)$ in $\mathbb{R}^d$, iid
- Response variables:

$$y_i = x_i^T \theta_* + \varepsilon_i$$

  where $\theta_* \in \mathbb{R}^d$ "true signal", $\varepsilon_i \sim N(0, \sigma^2)$, iid, so the overall noise is $\varepsilon \sim N(0, \sigma^2 I_n)$
- As usual we denote by $X \in \mathbb{R}^{n \times d}$ the design matrix with data points as rows, and by $\hat{\Sigma} = \frac{1}{n} X^T X \in \mathbb{R}^{d \times d}$ the empirical covariance matrix.
- The definition of the excess risk at test point x is defined as

$$R(\hat{\theta}) = E_x((\hat{\theta}^\top x - \theta_*^\top x)^2)$$

  The expectation is over the test point x. We will then consider the expectation with respect to the training data of the excess risk: $E_{X,y} R(\hat{\theta})$.

Setup: linear model, isotropic Gaussian sampling (2)

Observe that in this setting, there might not be enough data to learn all the model parameters:

- We get n scalar observations $y_1, ..., y_n$ from the n input points
- We need to learn all the d parameters of θ_*
- We assume $n \ll d$.
- Writing it up as a linear system we have d unknowns but only n equations.

Thus, in general we might not have enough data to perfectly identify the signal.

Main result: Expected excess risk in the over-parametrized regime

Theorem 13 (Expected excess risk, linear model, Gaussian data)

In the setup outlined in the previous slide, the expected excess risk is given as follows:

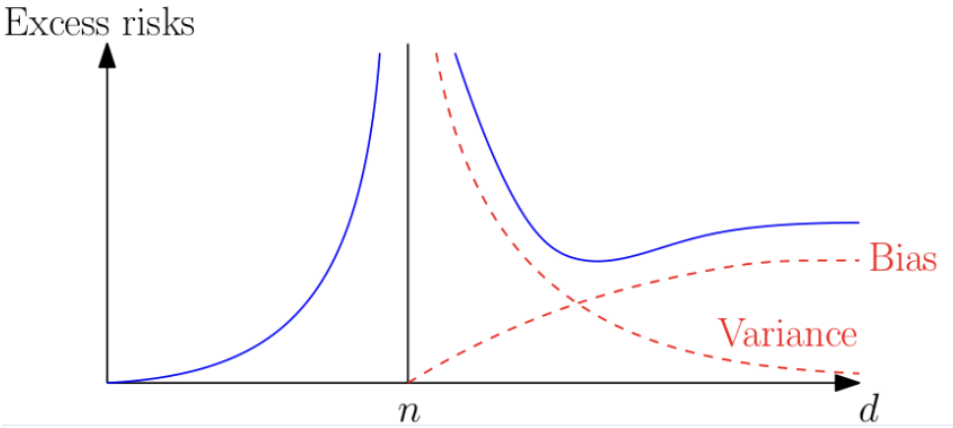
- In the under-parameterized regime $n > d + 1$, we have

$$E_{X,y}(R(\hat{\theta})) = \underbrace{\frac{\sigma^2 d}{n - d - 1}}_{\textit{variance}}$$

- In the over-parameterized regime $d > n + 1$ we have:

$$E_{X,y}(R(\hat{\theta})) = \underbrace{\frac{\sigma^2 n}{d - n - 1}}_{\textit{variance}} + \underbrace{\|\theta_*\|_2^2 \frac{d - n}{d}}_{\textit{bias}}$$

Main result: Expected excess risk in the over-parametrized regime (2)



Not a disaster, but learning in the underparameterized regime works better than in the overparameterized one!

Intuition for the underparameterized case

Variance:

- Consider a case where d is very small and fixed, and n is huge.
- Then we essentially average over n points to estimate the signal, and the variance of this scales as $1/n$.
- Thus the variance term decreases as $1/n$ because we average the noise over many observations.

Bias: We have enough data to see all the signal, so no systematic bias occurs.

Intuition close to interpolation threshold

Essentially we have about the same number of equations as unknowns, which is fine for fitting the training data without bias, but doesn't allow to exploit averaging to reduce the variance: it explodes on test points.

Generalization impossible in this regime.

Intuition for the overparameterized case

Variance:

- Consider a case where we have just 1 data point x, but a large number d of dimensions.
- The noise can be interpolated by making tiny changes across all of the d coordinates of $\hat{\theta}$. And this is exactly what happens with the minimum norm interpolator. And a fresh test point has only a small accidental alignment with the noise-fitting component, inducing variance of only size $1/d$.
- So intuitively, the variance decreases because we can "spread" the noise-fitting part of the parameter vector over many coordinates.
- More detailed, the minimum norm solution is

$$\hat{\theta} = y\frac{x}{\|x\|^2} = \underbrace{(x^T\theta_* + \varepsilon)}_{y}\frac{x}{\|x\|^2} = \underbrace{\frac{x^T\theta_*}{\|x\|^2}x}_{\text{signal part}} + \underbrace{\frac{\varepsilon}{\|x\|^2}x}_{\text{noise part}}.$$

Intuition for the overparameterized case (2)

- In our setting (isotropic sampling), x will have entries in all its coordinates, and $\|x\|^2$ is of the order d.
- Thus due to the noise part, every coordinate of $\hat{\theta}$ changes by order $1/d$ in response to the noise.
- For an independent test point, this boils down to variance of order $1/d$.

Intuition for the overparameterized case (3)

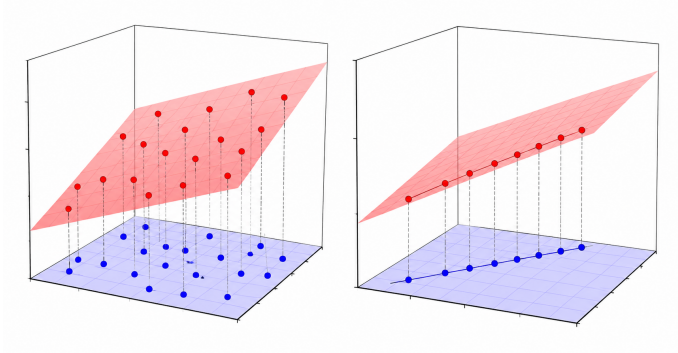
Bias:

- The n training points can only reveal the part of the target that lies in their n-dimensional span. The part in the remaining $d - n$ directions is completely unobserved.
- By symmetry of the sampling distribution, an n/d fraction of the signal lies within that subspace and can be observed (on average).
- The other way round, we are "missing" a $(d - n)/d$ fraction of the signal.

- (Compare this result to the previous one (Th. 12) where the target function was 0. In that case, there was nothing to miss: the minimum norm solution is close to 0 whereever we don't see data, and 0 was what we were supposed to predict. So the previous result had no bias term.)

Intuition for the overparameterized case (4)

Illustration for the bias: Blue: input data, red: true linear output signal



Left: The data covers all of the space, so we can estimate the slope of the red plane.
Right: Here, the data lives in a subspace (as it is always the case in the $n < d$ setting). So we can only estimate how the slope of the red plane behaves in this subspace, but not how it is tilted in directions orthogonal to it. This introduces a bias.

Proof step 1: excess risk

We first derive a simpler expression for the excess risk:

$$
\begin{aligned}
R(\hat{\theta}) &= E_x((\hat{\theta}^\top x - \theta_*^\top x)^2) \\
&= E_x\big((\hat{\theta}^\top x - \theta_*^\top x)^T(\hat{\theta}^\top x - \theta_*^\top x)\big) \\
&= (\hat{\theta} - \theta_*)^T \underbrace{E(xx^\top)}_{=:\Sigma}(\hat{\theta} - \theta_*) \\
&= (\hat{\theta} - \theta_*)^T\Sigma(\hat{\theta} - \theta_*) \\
&=: \|\hat{\theta} - \theta_*\|_\Sigma^2
\end{aligned}
$$

SKIPPED

Proof step 2: Bias-Variance- decomposition

Now additionally consider the expectation over the training points. In this setting, a version of the bias-variance-decomposition is given as

$$\begin{aligned} E_{X,y} R(\hat{\theta}) &= E_{X,y} \|\hat{\theta} - \theta_*\|_{\Sigma}^2 \\ &= E_{X,\varepsilon} \|\hat{\theta} - \theta_*\|_{\Sigma}^2 \\ &= \underbrace{E_X(\|E_\varepsilon(\hat{\theta}|X) - \theta_*\|_{\Sigma}^2)}_{\text{bias term}} + \underbrace{E_{X,\varepsilon}(\|\hat{\theta} - E_\varepsilon(\hat{\theta}|X)\|_{\Sigma}^2)}_{\text{variance term}} \end{aligned}$$

(We can replace the expectation wrt y by the one wrt to the noise variable ε because the response variable y is given as deterministic function plus noise;).

SKIPPED

Proof step 3: Analysis in the under-parameterized regime $n \geq d + 2$

The minimum-norm interpolator is simply the solution of ordinary least squares:

$$\hat{\theta} = (X^T X)^{-1} X^T y$$

- We have already seen the results regarding the expected excess risk:
  In the random design setting, the expected (wrt X, y) excess risk is given by

$$E_{X,y}(R(\hat{\theta})) = \frac{\sigma^2}{n} E_{X,y}\big(\mathrm{tr}(\underbrace{\Sigma}_{I_d} \underbrace{\hat{\Sigma}^{-1}}_{(\frac{1}{n}(X^T X))^{-1}})\big) = \sigma^2 E_X\big(\mathrm{tr}((X^T X)^{-1})\big)$$

SKIPPED

Proof step 3: Analysis in the under-parameterized regime $n \geq d + 2$ (2)

- The matrix $X^T X$ follows a Wishart distribution with n degrees of freedom. It is known that for $n \geq d + 2$ we have

$$E((X^T X)^{-1}) = \frac{1}{n - d - 1} I_d,$$

hence its expected trace is $d/(n - d - 1)$.

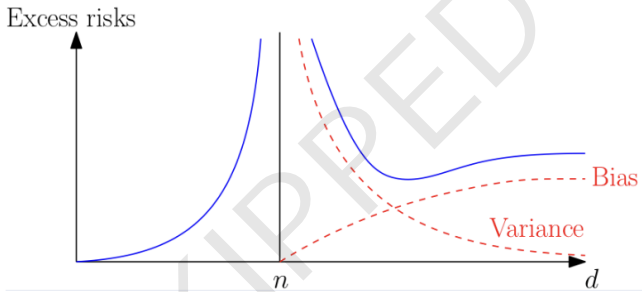
- So overall the excess risk in the under-parameterized regime is given as

$$E(R(\hat{\theta})) = \sigma^2 \frac{d}{n - d - 1}$$

SKIPPED

Proof step 3: Analysis in the under-parameterized regime $n \geq d + 2$ (3)

Observe that for $d \ll n$, the excess risk is small:



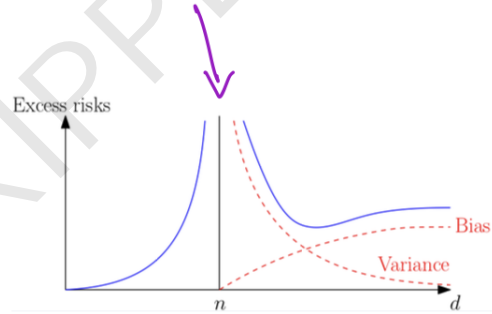
under-parameterized regime

SKIPPED

Proof step 4: What happens close to the interpolation threshold, $n \approx d$

- For $n = d$ and $n = d + 1$ it holds that $E(\mathrm{tr}((X^T X)^{-1})) = \infty$ (it is a property of Wishart matrices when the degree of freedom is d or $d + 1$).
- Hence, the variance explodes.
- This is why the double-descent-plot diverges close to the interpolation threshold $d = n$.

SKIPPED



Proof step 5: Analysis in the over-parametrized regime $n + 1 < d$

Before we get started with the maths in the over-parameterized settings, let us build some intuition:

- We have a signal θ_* in a d-dimensional space. But in the setting $n \ll d$ we have much less data points than dimensions.
- So even if each data point would serve to perfectly identify one of the d parameters of θ_*, we will in general NOT have enough data to observe all the d different directions of θ_*.
- So in general, we will NOT converge to a small risk!
- To achieve a small risk, we need additional assumptions. These assumptions would need to capture that the true signal θ_* can in fact be described by at most of the order n many parameters (not d many).

SKIPPED

Proof step 5: Analysis in the over-parametrized regime $n + 1 < d$ (2)

Let us analyze the situation formally:

- We consider the bias-variance-decomposition:

$$E_{X,\varepsilon}(R(\hat{\theta})) = \underbrace{E_{X,\varepsilon}(\|E_{\varepsilon}(\hat{\theta}|X) - \theta_*\|_{\Sigma}^2)}_{bias} + \underbrace{E_{X,\varepsilon}(\|\hat{\theta} - E_{\varepsilon}(\hat{\theta}|X)\|_{\Sigma}^2)}_{variance}$$

- We have already seen that in this scenario, GD/SGD converges to the minimum norm interpolator, which is given by

$$\hat{\theta} = X^T(XX^T)^{-1}y$$

SKIPPED

Proof step 6: Variance term in the over-parameterized regime

To evaluate the green part in the variance term, first observe that

$$\begin{aligned}\hat{\theta} &= X^T(XX^T)^{-1}y \\ &= X^T(XX^T)^{-1}(X\theta_* + \varepsilon) \\ &= X^T(XX^T)^{-1}X\theta_* + X^T(XX^T)^{-1}\varepsilon.\end{aligned}$$

And because of the independence of the noise, $E_\varepsilon(\varepsilon \mid X) = 0$. Thus

$$E_\varepsilon(\hat{\theta} \mid X) = X^T(XX^T)^{-1}X\theta_*.$$

So the green term is

$$\hat{\theta} - E_\varepsilon(\hat{\theta} \mid X) = X^T(XX^T)^{-1}\varepsilon.$$

SKIPPED

Proof step 6: Variance term in the over-parameterized regime (2)

Now we compute the full variance term:

$$
\begin{aligned}
E_{X,\varepsilon}\left(\ \|\hat{\theta} - E_{\varepsilon}(\hat{\theta} \mid X)\|_{\Sigma}{}^2 \ \right) &= E_{X,\varepsilon}(\|X^T(XX^T)^{-1}\varepsilon\|_{\Sigma}{}^2) \\
&= E_{X,\varepsilon}(\varepsilon^T(XX^T)^{-1}X\,\Sigma\,X^T(XX^T)^{-1}\varepsilon) \\
&\text{(exploit } \Sigma = I\text{)} \\
&= E_{X,\varepsilon}(\varepsilon^T(XX^T)^{-1}XX^T(XX^T)^{-1}\varepsilon) \\
&\text{(green terms cancel)} \\
&= E_{X,\varepsilon}\big(\varepsilon^T(XX^T)^{-1}\varepsilon\big) \\
&\text{(trace argument as before; exploit that conditional on } X \\
&\text{we have that } E_{\varepsilon}(\varepsilon\varepsilon^T) = \sigma^2 I_n\text{, which is independent of } X\text{)} \\
&= \sigma^2 E_X\big(\mathrm{tr}\,(XX^T)^{-1}\big) \\
&= \sigma^2 \frac{n}{d-n-1}
\end{aligned}
$$

SKIPPED

Proof step 6: Variance term in the over-parameterized regime (3)

The variance term behaves nice in the over-parameterized regime: it goes to 0 as $d \to \infty$ with $n/d \to 0$.

(Observe that it is the same variance term as we obtained in the toy setup with the constant-0 function above).

SKIPPED

Proof step 7: Bias term in the over-parameterized regime

$$\underbrace{E_{X,\varepsilon}\left(\|E_\varepsilon(\hat{\theta} \mid X) - \theta_*\|_\Sigma^2\right)}_{\text{bias term}} = E_X\left(\|E_\varepsilon(\hat{\theta} \mid X) - \theta_*\|_\Sigma^2\right)$$

$$= E_X\left(\left\|E_\varepsilon\left(X^T(XX^T)^{-1}(X\theta_* + \varepsilon)\,\middle|\, X\right) - \theta_*\right\|_\Sigma^2\right)$$

$$= E_X\left(\left\|X^T(XX^T)^{-1}X\theta_* + X^T(XX^T)^{-1}E_\varepsilon(\varepsilon \mid X) - \theta_*\right\|_\Sigma^2\right)$$

the green term is 0

$$= E_X\left(\|X^T(XX^T)^{-1}X\theta_* - \theta_*\|_\Sigma{}^2\right)$$

exploit that $\Sigma = I_d$

$$= E_X\left(\theta_*^T(I - X^T(XX^T)^{-1}X)\theta_*\right)$$

SKIPPED

Proof step 7: Bias term in the over-parameterized regime (2)

For the blue term: The matrix $X^T(XX^T)^{-1}X \in \mathbb{R}^{d\times d}$ is a projection matrix on a random subspace of dim $n < d$:

- A projection is a linear mapping that satisfies $P^2 = P$. It is easy to see that this holds here.
- One can also see that it maps every data point (row of X) to itself, and every vector of range(X) is being mapped to itself.
- For every vector $x \in (\text{range}(X))^{\perp} = \ker(X)$ one can see that it is mapped to 0.

SKIPPED

Proof step 7: Bias term in the over-parameterized regime (3)

Because our data comes from $N(0, I)$, the random subspace is uniformly distributed. So in the computation of the expected value we can replace θ_* with $\|\theta_*\| \cdot e_1$ where e_j is the j-th unit vector (formally, we rotate the data to achieve this). Thus we get:

$$
\begin{aligned}
E\left(\theta_*^\top X^T (XX^T)^{-1} X \theta_*\right) &\rightsquigarrow \text{replace } \theta_* \text{ with } \|\theta_*\| e_j \text{ for some } j \\
&= \|\theta_*\|_2^2 \cdot E\left(e_j^T X^T (XX^T)^{-1} X e_j\right) \\
&\rightsquigarrow \text{even do it for all } j \text{ and build the rescaled sum} \\
&= \|\theta_*\|_2^2 \cdot E\left(\frac{1}{d} \sum_{j=1}^{d} e_j^T X^T (XX^T)^{-1} X e_j\right) \\
&= \|\theta_*\|_2^2 \cdot \frac{1}{d} \sum_{j=1}^{d} E\left(e_j^T X^T (XX^T)^{-1} X e_j\right)
\end{aligned}
$$

SKIPPED

Proof step 7: Bias term in the over-parameterized regime (4)

$$\rightsquigarrow \text{exploit } \sum_{j=1}^{d} e_j^\top A e_j = tr(A)$$

$$= \|\theta_*\|_2^2 \cdot \frac{1}{d} E\big(\text{tr}\big(X^T (XX^T)^{-1} X\big)\big) =$$

$$= \|\theta_*\|_2^2 \cdot \frac{n}{d}$$

For the last equality, we exploit again that the matrix $X^T(XX^T)^{-1}X$ is the projection that takes any vector in $\mathbb{R}^d$ and projects it on the n-dim subspace span $x_1, ..., x_n = range(X^\top)$. Its trace is n.

Or alternatively, you can use the cyclicity of the trace directly: $\text{tr}(ABC) = \text{tr}(CAB)$.

SKIPPED

Proof step 7: Bias term in the over-parameterized regime (5)

So all in all, the bias term gives:

$$\begin{aligned} E_X\left(\theta_*^T\left(I - X^T(XX^T)^{-1}X\right)\theta_*\right) &= E_X\left(\theta_*^T\theta_*\right) - E\left(\theta_*^T(X^T(XX^T)^{-1}X)\theta_*\right) \\ &= \|\theta_*\|^2 - \|\theta_*\|^2\frac{n}{d} \\ &= \|\theta_*\|^2 \cdot \frac{d-n}{d} \end{aligned}$$

SKIPPED

Proof step 7: Bias term in the over-parameterized regime (6)

Let us discuss what this bias term means:

If $\|\theta_*\| = 0$, the bias is 0. This is great, and it corresponds to our toy example that we have seen before. We get benign overfitting.

If $\|\theta_*\| \neq 0$, the situation is bad:

- because we are in the over-parameterized regime $n \ll d$, the term $(d-n)/d \approx \Theta(1)$. It does not vanish, there is some bias that remains. We cannot recover the true underlying model!
- This is NOT a regime of benign overfitting in general! We interpolate, but there is a large bias that does not disappear!
- Intuitively, the reason is that with n points we can only capture n/d-fraction of the signal, the rest "gets lost"! (unless we make strong assumptions, see below).

SKIPPED

☺

Benign overfitting: Linear model with aligned covariance structure

Literature: Deep learning, a statistical viewpoint (Bartlett, Montanari, Rakhlin, Acta Numerica 2021)

More general setup: linear model, arbitrary sampling

- Assume input points x_i come from some distribution on $\mathbb{R}^d$ with $E(x_i) = 0$.
- Denote the true covariance matrix of the input distribution by Σ
- Consider the eigenvalues of the covariance matrix: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$
- W.l.o.g. we assume that the eigenvectors are the unit vectors e_1, ..., e_d
- Assume the true regression function is linear:

$$f^*(x) = \langle \theta_*, x \rangle \quad \text{for some } \theta_* \in \mathbb{R}^d.$$

- We consider the over-parameterized regime $d > n$.
- **We want to know whether/when we can really learn the linear model without any bias in this overparameterized setting.**

Need assumptions

- As observed before, there might not be enough data to learn all the model parameters (n equations for d unknowns). Thus, perfect learning will only be possible under additional assumptions.
- First, we need **assumptions on the "signal"**: we must be able to "express it" with much less than d "numbers", so that if we see n points, we have enough information to estimate the "numbers".
- Second, we need **assumptions on the sampling** mechanism: we need to make sure that the variance of the sampling distribution is aligned with the signal. See next pages for intuition.

Need assumptions (2)

Assumptions on the signal:

Want to estimate the vector $\theta_* = (\theta_{*1}, ..., \theta_{*d})^t$.

Possible assumptions that might work:

- The vector is (approximately) sparse and only contains few (not more than n) non-zero components.
- The vector might be approximately sparse if expressed in a different basis (for example, the basis given by the principal components of the data).
- The data lives on some low-dimensional manifold
- etc

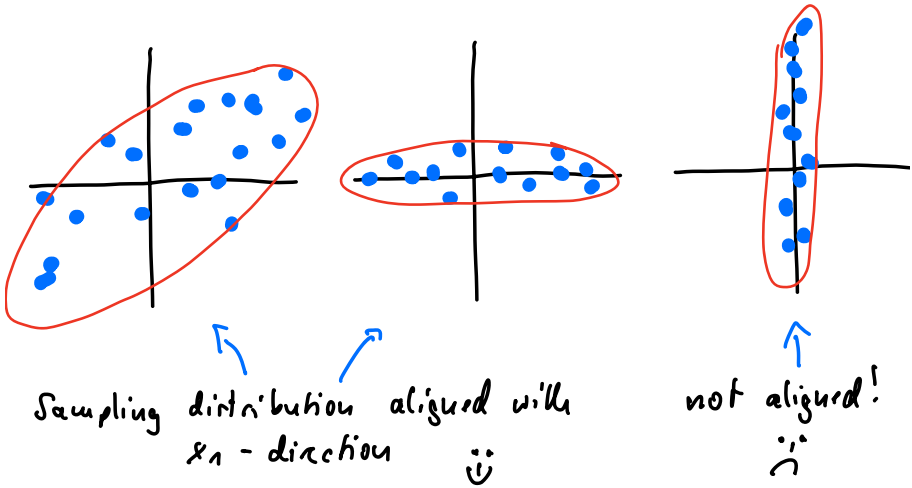
Need assumptions (3)

Assumptions on alignment:
Even if the signal is inherently low-dimensional, we need to make sure that we are able to observe the signal in all important directions.

Example:

- Two-dimensional setup, true linear parameter vector $\theta_* = (1, 0)^t$, so for $x = (x_1, x_2)^t \in \mathbb{R}^2$ the true function is $f(x) = {\theta_*}^t x = x_1$.
- Observing this signal will only be possible if we see enough data points with different x_1-components.
- We express this by saying that the covariance of the sampling distribution in direction of e_1 must be reasonably large.
- See illustration on next page.

Need assumptions (4)



Main results, informal

- We work in the basis given by the covariance matrix (see setup slide), and wlog we assume that the eigenvectors are $e_1, ..., e_d$ with eigenvalues are ordered such that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d$.
- **We now split the space in two subspaces**, the first one spanned by eigenvectors $e_1, \ldots, e_k$ and the second one spanned by $e_{k+1}, \ldots, e_d$ (for some choice of k).
- The idea of the analysis is to use the first group for the prediction part (the function we want to learn), while the second group takes care of interpolation (the spikes that are necessary to interpolate).

Main results, informal (2)

- Specifically, we choose k by

$$k = \min\left\{\ell \,\middle|\, \frac{\sum_{i>\ell} \lambda_i}{\lambda_{\ell+1}} \geq \text{const} \cdot n\right\}$$

  - The fraction compares the total variance in the tail starting at ℓ with the largest individual tail eigenvalue $\lambda_{\ell+1}$. If this term is large (at least of order n), then the tail variance is spread over many directions, rather than dominated by one or a few eigenvectors.
  - In particular, the tail is hopefully "heavy enough" to allow interpolation.
  - The exact form of this term is chosen such that the analysis goes through.
- With the help of this construction one can prove the following theorem:

Main results, informal (3)

Theorem 14 (Expected excess risk of minimum norm interpolator, general case)

In the setup outlined above, let $\hat{\theta} = X^T(XX^T)^{-1}y$ be the minimum-norm interpolator. Choose $k < d$ as on the previous slide (so that the tail is not dominated by only a few eigenvalues). Then, with high probability over the choice of the design matrix X, the expected excess risk can be bounded by

$$E_\varepsilon R(\hat{\theta}) \lesssim \underbrace{\sum_{i=1}^{d} \lambda_i \theta_{*i}^2 \left(\frac{\frac{1}{n}\sum_{j>k}\lambda_j}{\lambda_i + \frac{1}{n}\sum_{j>k}\lambda_j} \right)^2}_{\text{bias: loss from directions not learned}} + \underbrace{\sigma^2 \left(\frac{k}{n} + n\frac{\sum_{i>k}\lambda_i^2}{\left(\sum_{i>k}\lambda_i\right)^2} \right)}_{\text{variance: cost of fitting the noise}}.$$

The proof of this theorem is quite difficult, we skip it (you cannot find it in a text book, but in the papers by Bartlett et al.)

Let us try to understand what the theorem means.

Variance term, intuition

The variance term in the theorem is this one:

$$\sigma^2 \Bigg(\underbrace{\frac{k}{n}}_{\substack{\text{variance in} \\ \text{the signal itself}}} + \underbrace{n \frac{\sum_{i>k} \lambda_i^2}{\left(\sum_{i>k} \lambda_i\right)^2}}_{\text{tail weight from interpolation part}} \Bigg)$$

This term depends on the input covariance and the noise σ, but not on the true direction θ_*.

This term is small if we have many small tail eigenvalues, and large if we have few dominant tail eigenvalues (see next slide):

SKIPPED

Variance term, intuition (2)

To get a feeling for the behavior of the variance term:

- As one extreme case, assume $\lambda_{k+1} = \cdots = \lambda_d = a$. Then

$$n \frac{\sum_{i>k} \lambda_i^2}{\left(\sum_{i>k} \lambda_i\right)^2} = n \frac{(d-k)a^2}{\left((d-k)a\right)^2} = \frac{n}{d-k}$$

  For small k, and small n and large d, this is small. Good.

- The other way round, assume $\lambda_{k+1} = a$, and all others are 0. Then

$$n \frac{\sum_{i>k} \lambda_i^2}{\left(\sum_{i>k} \lambda_i\right)^2} = n \frac{a^2}{a^2} = n$$

  which does not vanish. Bad.

SKIPPED

Variance term, intuition (3)

My personal intuition for this behavior:

- If we have many small terms: we use the terms to memorize the noise, but we also use the terms to achieve a small norm of our solution (minimum norm interpolator). Because we have so many possibilities to interpolate, both goals can be achieved in a satisfactory way. In particular, the memorization part only contains small spikes and otherwise fits the 0 function and doesn't hurt generalization. The resulting norm of the minimum norm interpolator is small.
- If we have only few large terms in the tail, then these terms will be used to memorize the noise, but nothing else can be achieved. In particular, while we always find a minimum norm interpolator, this norm would be much higher than in the other regime.

SKIPPED

Bias term, intuition

The bias term in the theorem is given as

$$\sum_{i=1}^{d} \lambda_i \cdot (\theta_{*i})^2 \Big(\frac{\frac{1}{n} \sum_{j>k} \lambda_j}{\lambda_i + \frac{1}{n} \sum_{j>k} \lambda_j} \Big)^2$$

SKIPPED

Bias term, intuition (2)

- Case where the signal is well aligned with the top eigenvectors:
  - Assume that the covariance mainly lives in the first dimension: $\lambda_1 \gg \sum_{j>k} \lambda_j$ and $\sum_{j>k} \lambda_j$ is small.
  - Assume the signal is aligned perfectly with the first eigenvector, that is $\theta_{*1} = 1$ and for all $i > 1$, $\theta_{*i} = 0$.
  - Then in the sum, all terms except the first one disappear, and the first one is "small". So the overall bias is small. Good.

- Case where the signal is well aligned with the small tail eigenvalues (that is, $\theta_{1*} = \theta_{2*} = ... = \theta_{k*} = 0$ and the remaining θ_j are large)..
  - Then the first k terms in the sum disappear, and the remaining ones are large (because λ_i is small).
  - So the bias gets worse. Bad!

SKIPPED

Combining bias and variance terms

Good situation:

- Covariance spectrum contains few large eigenvalues $\lambda_1 \dots \lambda_k$ and then many tiny remaining eigenvalues.
- The signal θ_* is aligned with the top eigenvalues.

Then:

- Then the true signal and the data essentially live in (the same) k-dimensional subspace.
- Then both bias and variance are small.
- The top k directions can be used to represent the signal, similarly to the under-parameterized regime.
- The remaining $d - k$ directions serve to interpolate, but don't accumulate bias.

Benign overfitting!

Combining bias and variance terms (2)

Several possibilities for bad situations:

- If the tail spectrum has too much weight, bias accumulates. The interpolator only catches a fraction of n/d signal, but misses $1 - n/d$ of the signal. Moreover, the variance term does not go to 0. (This was the situation that we had already discussed earlier where we sampled the data from $N(0, I)$.
- If the tail spectrum contains few dominant eigenvalues, the variance does not go to 0.
- If the signal θ_* is not well-aligned with the dominating eigenvalues, then the bias term does not go to 0. Intuitively, we don't have enough data to learn the signal.

Different regimes

On the following slides we illustrate the behavior in a toy example:

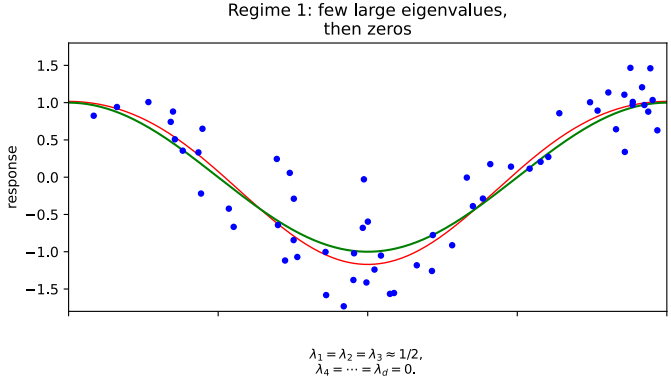
- $n = 60$ points sampled uniformly from $[0, 2\pi]$,
- We represent these points in a Fourier basis with the basis functions $\varphi_j(x) = s_j \cos(jx)$, in dimension $d = 2000$ (we will choose different values for s_j later).
- One can see that in this setup, the eigenvectors of the true covariance matrix are already the basis vectors, hence the true covariance matrix is a diagonal matrix. Its diagonal entries / eigenvalues are $\lambda_i = s_i^2/2$.
- As the true underlying function we choose $\cos(x)$ (the first basis function). This corresponds to a linear model in the Fourier space with true parameters $\theta_* = (1, 0, \ldots, 0)$.
- Observe that in this setup, the signal is always perfectly aligned with the first eigenvector.
- We sample the noisy output values $y_i = \cos(x_i) + \varepsilon_i$ with $\varepsilon_i \sim \mathcal{N}(0, 0.01)$
- Then we look at the minimum-norm interpolating solution (minimal least squares loss).

Different regimes (2)

- On the next slides we vary the eigenvalues of the covariance matrix. We always plot the true underlying function in green, the sampled data points in blue, and estimated function / minimum norm interpolator in red.

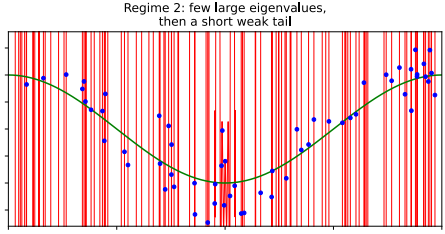
Different regimes (3)

1. **Few large eigenvalues, then many 0 eigenvalues:** similar to the under-parameterized regime. No overfitting possible, but learning works if signal is well-aligned with covariance structure.

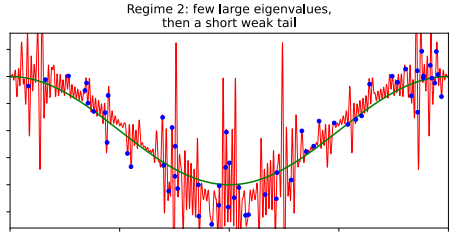


Different regimes (4)

2. **Few large eigenvalues, then few small ones:**
   Here interpolation is possible if we have enough small eigenvalues in the tail (essentially, we need n non-zero eigenvalues in the tail). But the variance remains large. We depict two cases, one where we just have enough non-zero eigenvalues to interpolate (left), and another case where we have some more non-zero eigenvalues (right). We can see that the variance of the red predicted function decreases with with the number of small non-zero eigenvalues.



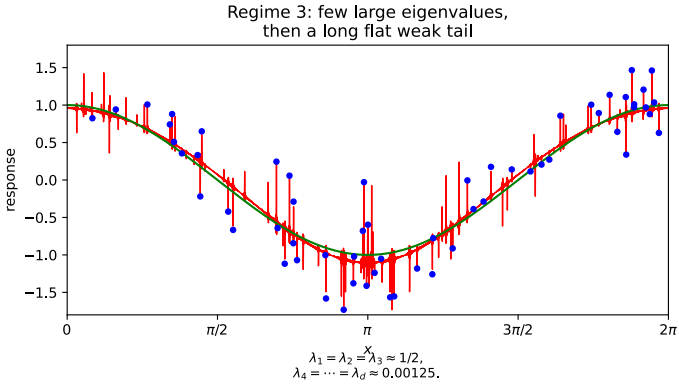
$\lambda_1 = \lambda_2 = \lambda_3 \approx 1/2,$
$\lambda_4 = \cdots = \lambda_{60} \approx 0.005,$
$\lambda_{61} = \cdots = \lambda_d = 0.$



$\lambda_1 = \lambda_2 = \lambda_3 \approx 1/2,$
$\lambda_4 = \cdots = \lambda_{150} \approx 0.005,$
$\lambda_{151} = \cdots = \lambda_d = 0.$

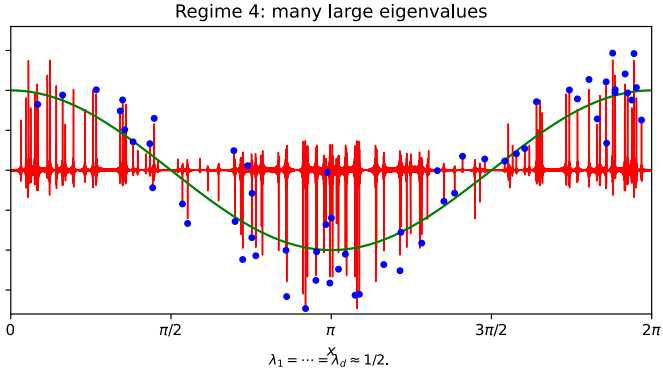
Different regimes (5)

3. **Few large eigenvalues, many small ones:** benign overfitting



Different regimes (6)

4. **Many large eigenvalues:**
   impossible to learn, bias remains high (this is essentially the situation as in the isotropic Gaussian sampling that we have seen already before)



Linear regression: summary

Whether benign overfitting can take place depends on the covariance matrix of the data and whether it is aligned with the signal θ_* that we try to recover.

It can take place if the "effective dimension" of the signal k is small compared to n and if the remaining dimensions contain isotropic small noise to allow for the interpolation.

Analysis is difficult, and little is known beyond linear setting.

Summary of the three lectures

Lecture 1: Why / when does machine learning work in the underparameterized regime?

- Underparameterized regime
- Theoretical algorithm (empirical risk minimization), where we do not consider the question how to find it or which of many minimizers might be selected
- Worst case approach: if we want to be on the safe side, how large a function class (how much "overfitting") can we afford?
- Generalization bounds measure the capacity / size of the underlying function class. If it is not too large, learning works.
- This theory is super-useful to understand the basics of machine learning in the underparameterized regime (as it is still important in many applications, in particular if we want interpretable solutions).
- This theory cannot explain why deep learning works: the bounds are not wrong, but meaningless.

Lectures 2 and 3: why can learning in the overparameterized regime work at all?

To explain the generalization capability of DNNs, we currently use the following chain of arguments:

- Consider the highly over-parameterized regime, high-dimensional setting.
- By the geometry of the loss landscape, we can find global optima (interpolating solutions) easily, for example by SGD (haven't talked about this in these lectures).
- Under certain conditions, through implicit regularization, this interpolating solution has nice properties (e.g. it is the minimum norm solution)
- For such nice solutions, one can prove that under certain assumptions we achieve generalization (e.g. through the "simple-plus-spiky" decomposition).