# Learning in high dimension: some insights from statistical physics

after Florent Krzakala's second lecture in les Houches:

https://florentkrzakala.com/files/leshouches2020/courses/florent

Aurélien Garivier

October 23$^{\mathrm{rd}}$, 2020

# Table of contents

# Statistical Machine Learning

# Classification in statistical learning

Input space $\mathcal{X} = [0,1]^{27\times 27}$, label set $\mathcal{Y} = \{0,\dots,9\}$

Sample $S_n = (X_1, Y_1), \dots, (X_n, Y_n)$ (MNIST: 70000 images + labels)

Rule (or *hypothese*) $f : \mathcal{X} \to \mathcal{Y}$

Classification algorithm $\mathcal{A}_n : \begin{array}{ccc} (\mathcal{X}\times\mathcal{Y})^n & \to & \mathcal{Y}^{\mathcal{X}} \\ S_n & \mapsto & \hat{f}_n \end{array}$

## Statistical Learning

Assumption: the data was and will be generated by a random mechanism:

$$S_n = (X_1, Y_1), \dots, (X_n, Y_n) \overset{iid}{\sim} P \quad \text{probability law on } \mathcal{X} \times \mathcal{Y}$$

Prediction $\hat{Y}_i = f(X_i)$ induces a *loss* $\ell\big(\hat{Y}_i, Y_i\big)$ ex: $\ell(\hat{Y}_i, Y_i) = \mathbb{1}\{\hat{Y}_i \neq Y_i\}$

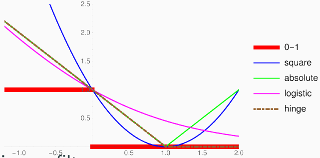
Oracle: rule $f$ minimizing $L(f) = \mathbb{E}_P\Big[\ell\big(f(X), Y\big)\Big]$ Minimizer = Bayes risk

Look for rules in hypotheses class $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$

For computational reasons (continuous optimisation is a lot easier than combinatorial optimization), one often considers relaxations:

- $\mathcal{Y}$ is taken to be convex Ex: $\{-1, 1\} \subset \mathbb{R}$, $\{0, \dots, 9\} \hookrightarrow \mathbb{R}^{10}$
- $\mathcal{F} = \big\{f_\theta : \theta \in \Theta \subset \mathbb{R}^p\big\}$ and consider $\text{sign}(f_\theta(X_i))$ as the prediction
- $\ell\big(\cdot, Y_i\big)$ = a smooth, convex function

  Ex: $\ell(\hat{Y}_i, Y_i) = (\hat{Y}_i - Y_i)^2$ or $\log_2\big(1 + \exp(-\hat{Y}_i \times Y_i)\big)$




Examples:

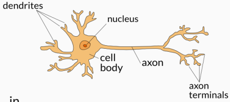
- linear classification $f_\theta(x) = x \cdot \theta$, $\theta \in \mathbb{R}^d$ ex: image filters
- neural networks $f_\theta = \boldsymbol{\sigma}_D \circ T_D \circ \dots \circ \boldsymbol{\sigma}_1 \circ T_1$

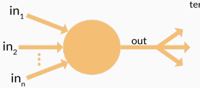
# Feedforward Neural Networks: Mimicking Brains?

**Neuron:** $x \mapsto \sigma\big(\langle w, x\rangle + b\big)$ with

- parameter $w \in \mathbb{R}^p$, $b \in R$
- (non-linear) activation function $\sigma : \mathbb{R} \to \mathbb{R}$

  typically $\sigma(x) = \frac{1}{1+\exp(-x)}$ or $\sigma(x) = \max(x, 0)$ called ReLU



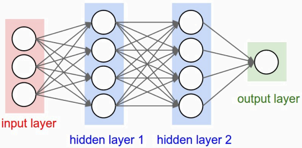

**Layer:** $x \mapsto \boldsymbol{\sigma}\big(Mx + \mathbf{b}\big)$ with

- parameter $M \in M_{q,p}(\mathbb{R}), \mathbf{b} \in \mathbb{R}^q$
- component-wise activation function $\boldsymbol{\sigma} = \sigma^{\otimes q}$

**Network**: composition of layers $f_\theta = \boldsymbol{\sigma}_D \circ T_D \circ \cdots \circ \boldsymbol{\sigma}_1 \circ T_1$ with

- architecture $A = \big(D, (p_1, \ldots, p_{D-1})\big)$
- $x_0 = x, \quad x_d = \boldsymbol{\sigma}_d\big(T_d x_{d-1}\big) \in \mathbb{R}^{p_d}$
- $T_d x = M_d x + \mathbf{b}_d$
- parameter $\theta = (M_1, \mathbf{b}_1, \ldots, \ldots, M_D, \mathbf{b}_D)$

  $\theta \in \Theta_A = \prod_{d=1}^{D} \mathcal{M}_{p_{d-1},p_d}(\mathbb{R}) \times \mathbb{R}^{p_d}$
- <u>*depth*</u> $D$ (⚠st. nb layers), <u>*width*</u> $\max_{1\leq d\leq D} p_d$



# Empirical Risk Minimization

Goal: find $\theta$ minimizing $L(\theta) = \mathbb{E}_P\Big[\ell\big(f_\theta(X), Y\big)\Big]$

But the learnt rule $\hat{f}_n = f_{\hat{\theta}_n}$ depends only on the sample $S_n$

PAC learning: for every $\epsilon, \delta > 0$, find the *sample size* $n(\epsilon, \delta)$ such that *whatever the law P*, if $n \geq n(\delta, \epsilon)$ then with probability at least $1 - \delta$ one has $L(\hat{\theta}_n) < \min_{\theta\in\Theta} L(\theta) + \epsilon$

Idea: Empirical Risk Minimization (ERM):

$$\hat{\theta}_n = \arg\min_{\theta\in\Theta} L_n(\theta) := \frac{1}{n}\sum_{i=1}^{n} \ell\big(f_\theta(X_i), Y_i\big)$$

PAC learning theory (pessimistic): uniform law of large numbers

$$\mathbb{P}\left(\forall\theta \in \Theta,\ \ \big|L_n(\theta) - L(\theta)\big| \leq c\sqrt{\frac{\dim\Theta + \log\frac{1}{\delta}}{n}}\right) \geq 1 - \delta$$

# Bias-variance tradeoff

The classical PAC theory does not work unless $n \gg \dim \Theta$

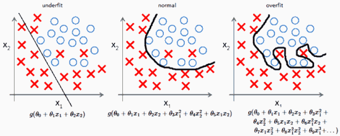
Consider different models $(\Theta_d)_{d\geq 1}$ Example: images at different resolutions

Decomposition of the (quadratic) risk

$$\mathbb{E}\left[L(\hat{\theta}_n)\right] = b_d^2 + v_d$$

- Bias: $b_d = \min_{\theta\in\Theta_d} L(\theta)$ decreases with $d$
- Variance term: $v_d = \frac{\dim \Theta_d}{n}$ increases with $d$

$\implies$ best choice = bias-variance balance




think: polynomial regression


src: `https://miro.medium.com`

# Bias-variance tradeoff

The classical PAC theory does not work unless $n \gg \dim \Theta$

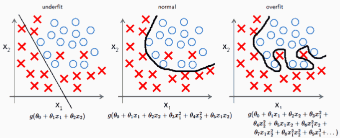
Consider different models $(\Theta_d)_{d \geq 1}$  Example: images at different resolutions

Decomposition of the (quadratic) risk

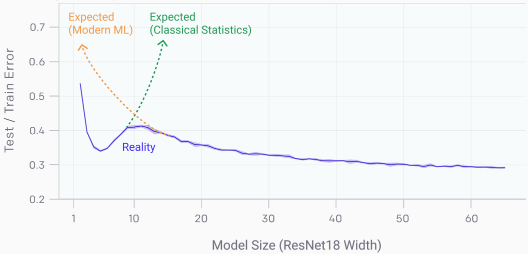
$$\mathbb{E}\left[L(\hat{\theta}_n)\right] = b_d^2 + v_d$$

- Bias: $b_d = \min_{\theta \in \Theta_d} L(\theta)$  decreases with $d$
- Variance term: $v_d = \frac{\dim \Theta_d}{n}$  increases with $d$

$\Longrightarrow$ best choice = bias-variance balance  think: polynomial regression




This statement is challenged by numerical experiments on neural networks




src: `https://miro.medium.com`

# Linear Models

# Linear models

$\hat{Y} = X \cdot \theta,\ \theta \in \mathbb{R}^d$

Matrix notation: $\mathbf{X} = \begin{pmatrix} X_1 \\ \vdots \\ X_n \end{pmatrix} \in M_{n,d}(\mathbb{R}),\ Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix} \in \mathbb{R}^n$

$L^2$ loss $\ell(\hat{Y}_i, Y_i) = (\hat{Y}_i - Y_i)^2$ solution: Ordinary Least Square (OLS)

$\hat{\theta}_n \in \underset{\theta \in \mathbb{R}^d}{\arg\min} \left\| Y - X\theta \right\|_2^2$ satisfies the normal equations $XX^T\theta = X^TY$

**If** $rank(X^TX) = d$ **(requires** $n \geq d$**)**

there is a unique solution

$$\hat{\theta}_n = (X^TX)^{-1}X^TY$$

Classical statistics theory

**Otherwise, many solutions**

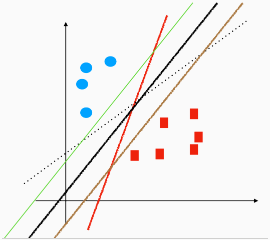
Popular: *least-norm* solution

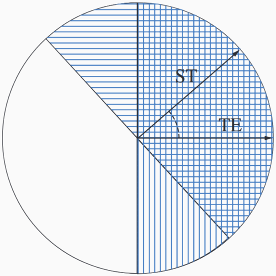
$$\hat{\theta}_n = X^T(XX^T)^{-1}Y$$

Statistically spurious

# Physics model: the teacher-student framework

- The model is true: $Y_i = \text{sign}(X_i \cdot \theta^*)$
- $X_i \sim \mathcal{N}(0, I_d)$ cf images?
- $\theta^* \sim \mathcal{N}(0, I_d)$ cf Bayesian approach?
- High-dimensional limit as $n, d \to \infty$ with $\alpha = n/d$ fixed






src: *Learning to generalize*, Opper'01

• volume of students with generalization error $\epsilon$: $v(\epsilon) \propto \epsilon^{d \times \text{entropy}(\epsilon)}$

• probability that a student with generalization error $\epsilon$ makes no mistake: $p(\epsilon) \propto \epsilon^{n \times \text{energy}(\epsilon)}$

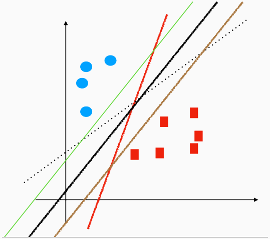
$\Longrightarrow$ average generalization error of ERM:

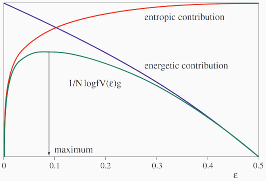
$$\int v(\epsilon)p(\epsilon)d\epsilon \approx \arg\max_{\epsilon} \text{entropy}(\epsilon) + \alpha\, \text{energy}(\epsilon)$$

in the limit when $\alpha = n/d$ fixed

# Physics model: the teacher-student framework

- The model is true: $Y_i = \text{sign}(X_i \cdot \theta^*)$
- $X_i \sim \mathcal{N}(0, I_d)$ cf images?
- $\theta^* \sim \mathcal{N}(0, I_d)$ cf Bayesian approach?
- High-dimensional limit as $n, d \to \infty$ with $\alpha = n/d$ fixed






src: *Learning to generalize*, Opper'01

• volume of students with generalization error $\epsilon$: $v(\epsilon) \propto \epsilon^{d \times \text{entropy}(\epsilon)}$

• probability that a student with generalization error $\epsilon$ makes no mistake: $p(\epsilon) \propto \epsilon^{n \times \text{energy}(\epsilon)}$

$\implies$ average generalization error of ERM:

$$\int v(\epsilon) p(\epsilon) d\epsilon \approx \arg\max_{\epsilon} \text{entropy}(\epsilon) + \alpha\, \text{energy}(\epsilon)$$

in the limit when $\alpha = n/d$ fixed

# Statistical physics analysis

$\Longrightarrow$ "spin glasses", physics analysis in the 1990's (Opper & Kinzel, etc.)
rigorous proofs recently (Florent Krzakala, Lenka Zdeborova, etc.):
can compute

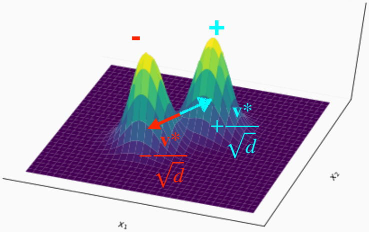
- the "Bayes risk" $\mathbb{E}\left[L(\theta)|S_n\right]$ = mean risk of ERM
- the risk of the minimal-norm ERM

in the limit, as a function of $\alpha$.

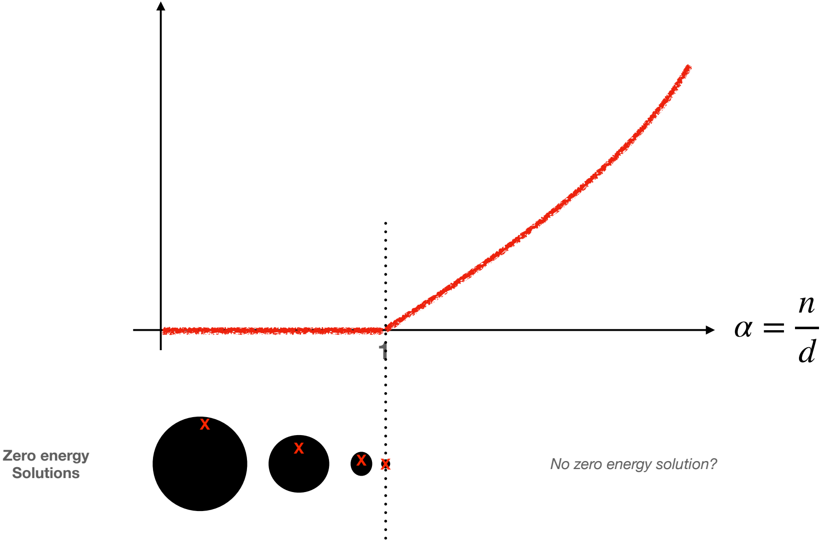
Other model discussed: Gaussian Mixture

$$X_i \sim \mathcal{N}\left(\frac{Y_i \mathbf{v}^*}{\sqrt{d}}, \Delta\right)$$

Similar results

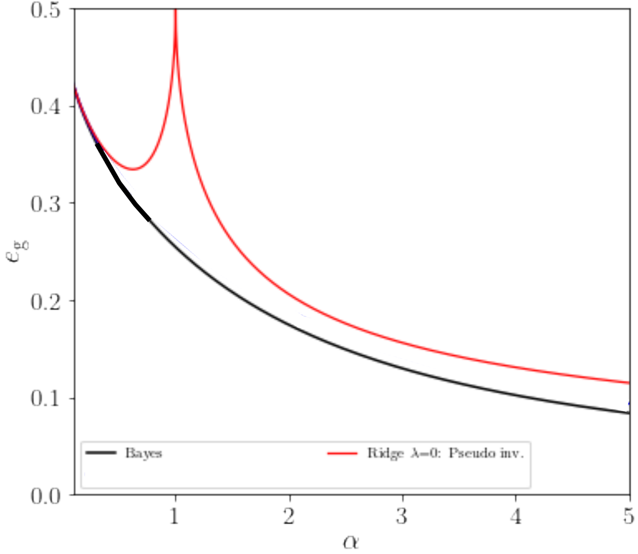


# Least norm solution




slide by Florent Krzakala

# Least norm solution: “double descent”




*slide by Florent Krzakala*

# Gradient Descent

Iterative optimization of $\theta$:

$$\theta^t = \theta^{t-1} - \eta_t \nabla_\theta L_n(\theta^{t-1})$$

Here $f_\theta(x) = x \cdot \theta \implies$

$$\nabla_\theta L_n(\theta^{t-1}) = \frac{1}{n}\sum_{i=1}^{n} \partial_1 \ell\big(f_\theta(X_i), Y_i\big) X_i \in \mathrm{span}(X_1, \ldots, X_n)$$

**Representer theorem**

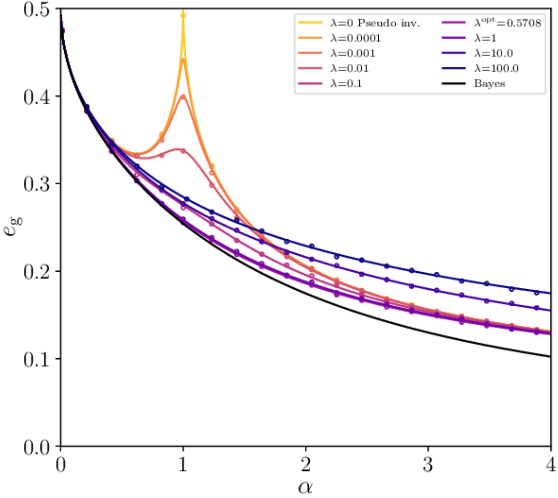
$$\min_{\theta \in \mathbb{R}^d} L_n(\theta) = \min_{\theta \in \mathrm{span}(X_1,\ldots,X_n)} L_n(\theta)$$

In fact, if $\theta = \theta_X + \theta_{X^\perp}$, then $L_n(\theta) = \frac{1}{n}\sum_{i=1}^n \ell\big(X_i \cdot \theta, Y_i\big) = L_n(\theta_X)$

$\implies$ if $\theta^0_{X^\perp} = 0$ gradient descent finds the solution with minimal norm

$\implies$ other approach: minimize *ridge loss* $L_n^\lambda(\theta) = L_n(\theta) + \lambda\|\theta\|^2$

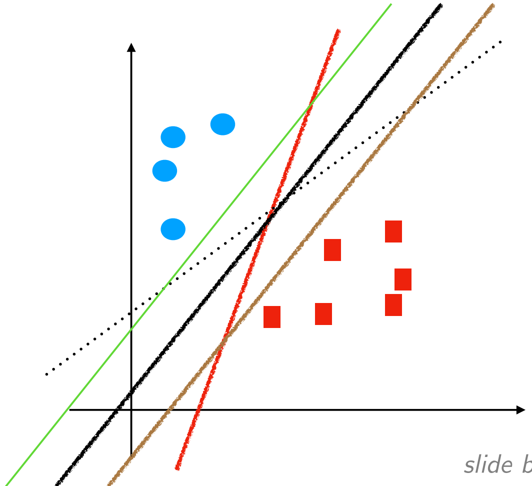
# Ridge loss




[Aubin, Krzakala, Lu, Zdeborova'20]

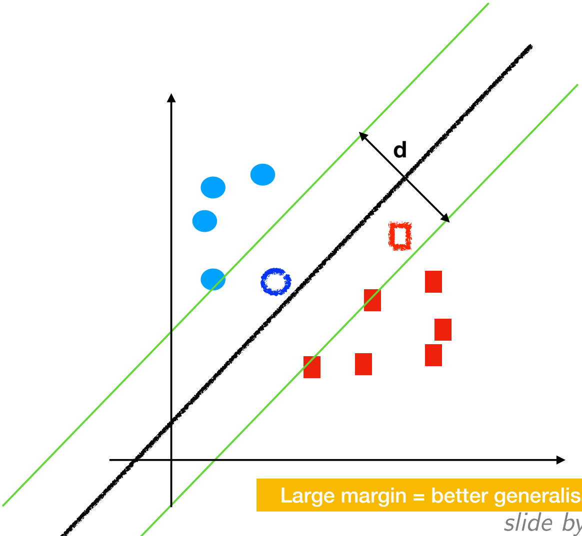
# Pushing the boundaries

**Which frontier should we choose?**



*slide by Florent Krzakala*

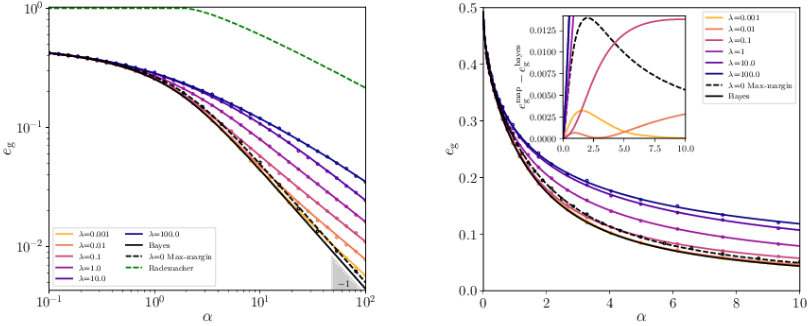
# Pushing the boundaries




Large margin = better generalisation properties!

*slide by Florent Krzakala*

# Chasing the Bayes optimal result




**Regularized logistic losses (almost) achieve Bayes optimal results!**

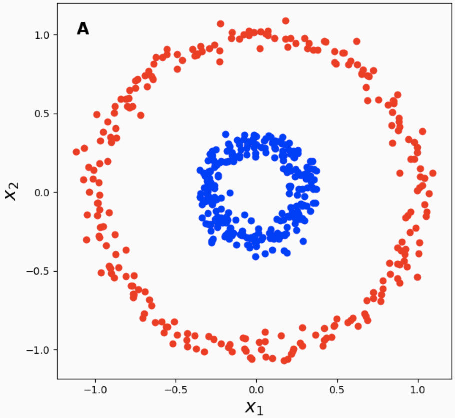
[Aubin, Krzakala, Lu, Zdeborova'20]

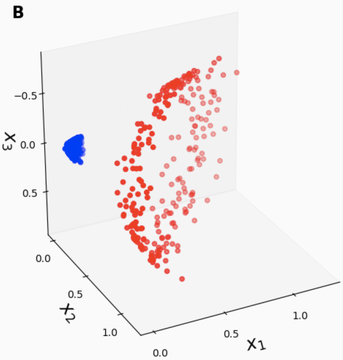
# Beyond Linear Models

# Lifting the data: feature map

Find a better representation of the data that makes it linearly separable

$$X_i \mapsto \Phi(X_i)$$




$$\Phi(x_1, x_2) = \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$$

src: http://gregorygundersen.com/

# Template matching

Representer theorem: can search for

$$\hat{\theta}_n = \sum_{i=1}^{n} \beta_i X_i$$

The resulting prediction is hardly more than comparison with data:

$$f_{\hat{\theta}_n}(x) = \sum_{i=1}^{n} \beta_i X_i \cdot x$$

(cf nearest neighbor method)

$\implies$ can consider more general similarity functions than scalar product:

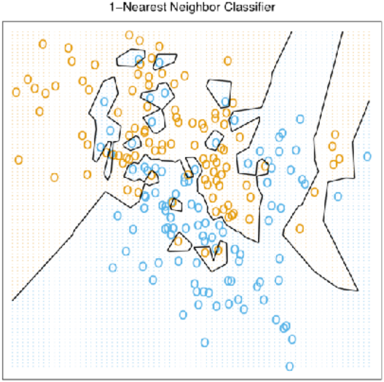
$$f_\theta(x) = \sum_{i=1}^{n} \beta_i \, K(X_i, x)$$

where $K$ is an carefully chosen *kernel*

# Ex: Gaussian Kernel

$$K(\mathbf{X}_i, \mathbf{X}_j) = e^{-\beta \|\mathbf{X}_i - \mathbf{X}_j\|_2^2}$$

**As $\beta \to \infty$ converges to 1NN methods**



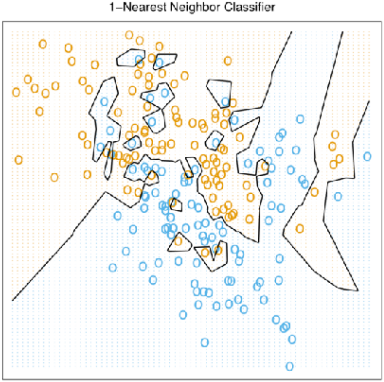

Warning: $\beta$ = inverse temperature $\neq$ the one of previous slide

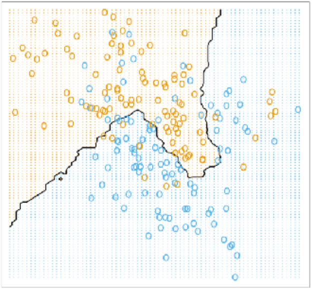
*slide by Florent Krzakala*

# Ex: Gaussian Kernel

$$K(\mathbf{X}_i, \mathbf{X}_j) = e^{-\beta \|\mathbf{X}_i - \mathbf{X}_j\|_2^2}$$

**For lower values, interpolate between neighbours**



Warning: $\beta$ = inverse temperature $\neq$ the one of previous slide

*slide by Florent Krzakala*

# Mercer's Theorem & the feature map

If K(s,t) is symmetric and positive-definite, then there is an orthonormal basis $\{e_i\}_i$ of $L^2[a, b]$ consisting of « **eigenfunctions** » such that the corresponding sequence of eigenvalues $\{\lambda_i\}_i$ is nonnegative.

$$K(s,t) = \sum_{j=1}^{\infty} \lambda_j \, e_j(s) \, e_j(t)$$

**All symmetric positive-definite Kernels can be seen as a projection in an infinite dimensional space**

**Orignal space (data space) dimension d**

**Feature map**

$$\Phi = g(X)$$

**Features space (After projection) dimension D (*possibly infinite***

$$X_i$$

$$K(\mathbf{X}_i, \mathbf{X}_j) = \mathbf{\Phi}_i \cdot \mathbf{\Phi}_j$$

$$\Phi_i = \begin{pmatrix} \sqrt{\lambda_1} e_1(X_i) \\ \sqrt{\lambda_2} e_2(X_i) \\ \sqrt{\lambda_3} e_4(X_i) \\ \cdots \\ \sqrt{\lambda_D} e_D(X_i) \end{pmatrix}$$

*slide by Florent Krzakala*

# Example: Gaussian Kernel, 1D

$$K(\mathbf{X}_i, \mathbf{X}_j) = e^{-\frac{1}{2\sigma^2}\|\mathbf{X}_i - \mathbf{X}_j\|_2^2}$$

$$e^{\frac{-1}{2\sigma^2}(x_i - x_j)^2} = e^{\frac{-x_i^2 - x_j^2}{2\sigma^2}} \left(1 + \frac{2x_i x_j}{1!} + \frac{(2x_i x_j)^2}{2!} + \ldots\right) \qquad (1.25)$$

$$= e^{\frac{-x_i^2 - x_j^2}{2\sigma^2}} \left(1 \cdot 1 + \sqrt{\frac{2}{1!}} x_i \cdot \sqrt{\frac{2}{1!}} x_j + \sqrt{\frac{(2)^2}{2!}} (x_i)^2 \cdot \sqrt{\frac{(2)^2}{2!}} (x_j)^2 + \ldots\right)$$

$$= \phi(x_i)^T \, \phi(x_j)$$

$$\text{where, } \phi(x) = e^{\frac{-x^2}{2\sigma^2}} \left(1, \sqrt{\frac{2}{1!}} x, \sqrt{\frac{2^2}{2!}} x^2, \ldots\right)$$

**Infinite dimensional feature (polynomial) map!**

*slide by Florent Krzakala*

# Kernel methods

$$\mathcal{R} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y_i, f_\beta(\mathbf{X}_i))$$

$$f_\beta(\mathbf{X}) = \sum_{j=1}^{n} \beta_j K(\mathbf{X}_j, \mathbf{X}) \quad \beta \in \mathbb{R}^n$$

Gradient descent

$$\beta^t = \beta^{t-1} - \eta \nabla_\beta \mathcal{R}$$

Gradient flow

$$\dot{\beta}^t = -\nabla_\beta \mathcal{R}$$

$$\mathcal{R} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y_i, f_\theta(\mathbf{\Phi}_i))$$

**Feature map** $\Phi = g(X)$

$$f_\theta(\mathbf{\Phi}) = \theta \cdot \mathbf{\Phi} \qquad \theta \in \mathbb{R}^{D=\infty}$$

Gradient descent

$$\theta^t = \theta^{t-1} - \eta \nabla_\theta \mathcal{R}$$

Gradient flow

$$\dot{\theta}^t = -\nabla_\theta \mathcal{R}$$

*slide by Florent Krzakala*

$$K(X_i, X_j) = \Phi_i \cdot \Phi_j$$

$$\mathbf{K} = \begin{pmatrix} K(X^1, X^1) & K(X^1, X^2) & \dots & K(X^1, X^N) \\ K(X^2, X^1) & K(X^2, X^2) & \dots & K(X^2, X^N) \\ \dots & \dots & \dots & \dots \\ K(X^N, X^1) & K(X^N, X^2) & \dots & K(X^N, X^N) \end{pmatrix}$$

**Say you have one million examples....**


*slide by Florent Krzakala*

# Kernel methods

$$\mathcal{R} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y_i, f_\beta(\mathbf{X}_i))$$

$$f_\beta(\mathbf{X}) = \sum_{j=1}^{n} \beta_j K(\mathbf{X}_j, \mathbf{X}) \qquad \beta \in \mathbb{R}^n$$

Gradient descent

$$\beta^t = \beta^{t-1} - \eta \nabla_\beta \mathcal{R}$$

Gradient flow

$$\dot{\beta}^t = -\nabla_\beta \mathcal{R}$$

$$\mathcal{R} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y_i, f_\theta(\mathbf{\Phi}_i))$$

**Feature map** $\Phi = g(X)$

$$f_\theta(\mathbf{\Phi}) = \theta \cdot \mathbf{\Phi} \qquad \theta \in \mathbb{R}^{D=\infty}$$

Gradient descent

$$\theta^t = \theta^{t-1} - \eta \nabla_\theta \mathcal{R}$$

Gradient flow

$$\dot{\theta}^t = -\nabla_\theta \mathcal{R}$$

*slide by Florent Krzakala*

# Kernel methods

$$\mathscr{R} = \frac{1}{n}\sum_{i=1}^{n} \mathscr{L}(y_i, f_\theta(\mathbf{\Phi}_i)) \qquad f_\theta(\mathbf{\Phi}) = \theta \cdot \mathbf{\Phi}$$

**Feature map** $\Phi = g(X)$

$$\theta \in \mathbb{R}^{D=\infty}$$

Gradient descent

$$\theta^t = \theta^{t-1} - \eta \nabla_\theta \mathscr{R}$$

Gradient flow

$$\dot{\theta}^t = -\nabla_\theta \mathscr{R}$$

Idea 1: truncate the expansion
of the feature map
(e.g. polynomial features)

Idea 2: approximate the
feature map by sampling

*slide by Florent Krzakala*

# Random Fourier Features [Recht-Rahimi '07]

Take $F_1, \dots, F_D \overset{iid}{\sim} Q$ Fourier coefficients in $\mathbb{R}^d$ and choose feature map

$$\Phi(x) = \frac{1}{\sqrt{D}} \begin{pmatrix} e^{i\,F_1 \cdot x} \\ \vdots \\ e^{i\,F_D \cdot x} \end{pmatrix}$$

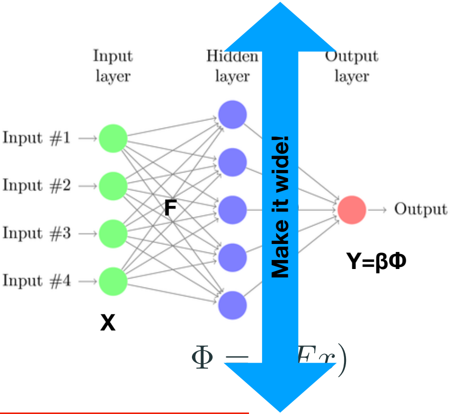
Then

$$K(X_j, X_k) = \Phi(X_j) \cdot \Phi(X_k) = \frac{1}{D} \sum_{\ell=1}^{D} e^{i\,F_\ell \cdot (X_j - X_k)} \overset{D \to \infty}{\longrightarrow} \int e^{i\,f \cdot (X_j - X_k)} dQ(f)$$

$\rightarrow$ if $dQ/df$ = Fourier transform of $\kappa$, then $K(X_j, X_k) \approx \kappa(X_j - X_k)$

| Kernel | $\kappa(\Delta)$ | $dQ(f)$ |
|---|---|---|
| Gaussian | $e^{-\|\Delta\|^2/2}$ | $(2\pi)^{-d/2} e^{-\|f\|^2/2}$ |
| Laplacian | $e^{-\|\Delta\|_1}$ | $\prod_k \frac{1}{\pi(1+f_k^2)}$ |
| Cauchy | $\prod_k \frac{2}{1+\Delta_k^2}$ | $e^{-\|f\|_1}$ |

# Equivalent representation:
## A WIIIIIIIDE random 2-layer neural network




Fix the « weights » in the first layer randomly…
… and to learn only the weights in the second layer

Infinitly wide neural net with random weights converges to kernel methods
( Neal ’96, Williams 98, Recht-Rahimi ‘07)

Deep connection with genuine neural networks in the “Lazy regime”
[Jacot, Gabriel, Hongler ’18; Chizat, Bach ’19; Geiger et al. ’19]

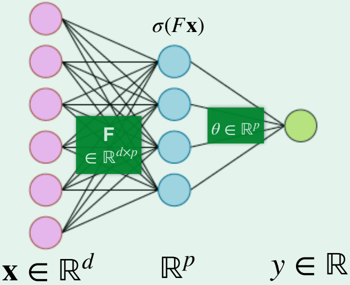
slide by Florent Krzakala

# Random feature model…

**Dataset:**

- n vector $\mathbf{x}_i \in \mathbb{R}^d$, drawn randomly from $\mathcal{N}(0,\mathbf{1}_d)$
- n labels $y_i$ given by a function $y_i^0 = f^0(\mathbf{x} \cdot \theta^*)$

**Architecture:** Two-layers neural network with fixed first layer **F**




**Cost function:**

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, y_i^0) + \lambda \|\theta\|_2^2 \qquad \ell(\,.\,) = $$

*Logistic loss*
*Hinge loss*
*Square loss*
…

**What is the training error & the generalisation error in the high dimensional limit $(d,p,n) \to \infty$?**

*slide by Florent Krzakala*

# … and its solution

Mention COUILLET HERE

[Loureiro, Gerace, **FK**, Mézard, Zdeborova, '20]

Definitions:

*Consider the unique fixed point of the following system of equations*

$$\begin{cases} \hat{V}_s = \frac{\alpha}{\gamma}\kappa_1^2 \mathbb{E}_{\xi,y}\left[\mathcal{Z}(y,\omega_0)\frac{\partial_\omega \eta(y,\omega_1)}{V}\right], \\ \hat{q}_s = \frac{\alpha}{\gamma}\kappa_1^2 \mathbb{E}_{\xi,y}\left[\mathcal{Z}(y,\omega_0)\frac{\left(\eta(y,\omega_1)-\omega_1\right)^2}{V^2}\right], \\ \hat{m}_s = \frac{\alpha}{\gamma}\kappa_1 \mathbb{E}_{\xi,y}\left[\partial_\omega \mathcal{Z}(y,\omega_0)\frac{\left(\eta(y,\omega_1)-\omega_1\right)}{V}\right], \\ \hat{V}_w = \alpha\kappa_\star^2 \mathbb{E}_{\xi,y}\left[\mathcal{Z}(y,\omega_0)\frac{\partial_\omega \eta(y,\omega_1)}{V}\right], \\ \hat{q}_w = \alpha\kappa_\star^2 \mathbb{E}_{\xi,y}\left[\mathcal{Z}(y,\omega_0)\frac{\left(\eta(y,\omega_1)-\omega_1\right)^2}{V^2}\right], \end{cases}$$

$$\begin{cases} V_s = \frac{1}{\hat{V}_s}\left(1 - z\, g_\mu(-z)\right), \\ q_s = \frac{\hat{m}_s^2+\hat{q}_s}{\hat{V}_s}\left[1-2zg_\mu(-z)+z^2 g_\mu'(-z)\right] \\ \quad -\frac{\hat{q}_w}{(\lambda+\hat{V}_w)\hat{V}_s}\left[-zg_\mu(-z)+z^2 g_\mu'(-z)\right], \\ m_s = \frac{\hat{m}_s}{\hat{V}_s}\left(1 - z\, g_\mu(-z)\right), \\ V_w = \frac{\gamma}{\lambda+\hat{V}_w}\left[\frac{1}{\gamma}-1+zg_\mu(-z)\right], \\ q_w = \gamma\frac{\hat{q}_w}{(\lambda+\hat{V}_w)^2}\left[\frac{1}{\gamma}-1+z^2 g_\mu'(-z)\right], \\ \quad +\frac{\hat{m}_s^2+\hat{q}_s}{(\lambda+\hat{V}_w)\hat{V}_s}\left[-zg_\mu(-z)+z^2 g_\mu'(-z)\right], \end{cases}$$

$$\begin{cases} \eta(y,\omega) = \underset{x\in\mathbb{R}}{\operatorname{argmin}}\left[\frac{(x-\omega)^2}{2V} + \ell(y,x)\right] \\ \mathcal{Z}(y,\omega) = \int \frac{dx}{\sqrt{2\pi V^0}} e^{-\frac{1}{2V^0}(x-\omega)^2}\delta\left(y - f^0(x)\right) \end{cases}$$

where $V = \kappa_1^2 V_s + \kappa_\star^2 V_w$, $V^0 = \rho - \frac{M^2}{Q}$, $Q = \kappa_1^2 q_s + \kappa_\star^2 q_w$, $M = \kappa_1 m_s$, $\omega_0 = M/\sqrt{Q}\xi$, $\omega_1 = \sqrt{Q}\xi$ and $g_\mu$ is the Stieltjes transform of $FF^T$

$\kappa_0 = \mathbb{E}\left[\sigma(z)\right], \kappa_1 \equiv \mathbb{E}\left[z\sigma(z)\right], \kappa_\star \equiv \mathbb{E}\left[\sigma(z)^2\right] - \kappa_0^2 - \kappa_1^2$ and $\vec{z}^\mu \sim \mathcal{N}(\vec{0}, \mathbf{I_p})$

Then in the high-dimensional limit:

$$\epsilon_{gen} = \mathbb{E}_{\lambda,\nu}\left[(f^0(\nu) - \hat{f}(\lambda))^2\right]$$

with $(\nu,\lambda) \sim \mathcal{N}\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}\rho & M^\star \\ M^\star & Q^\star\end{pmatrix}\right)$

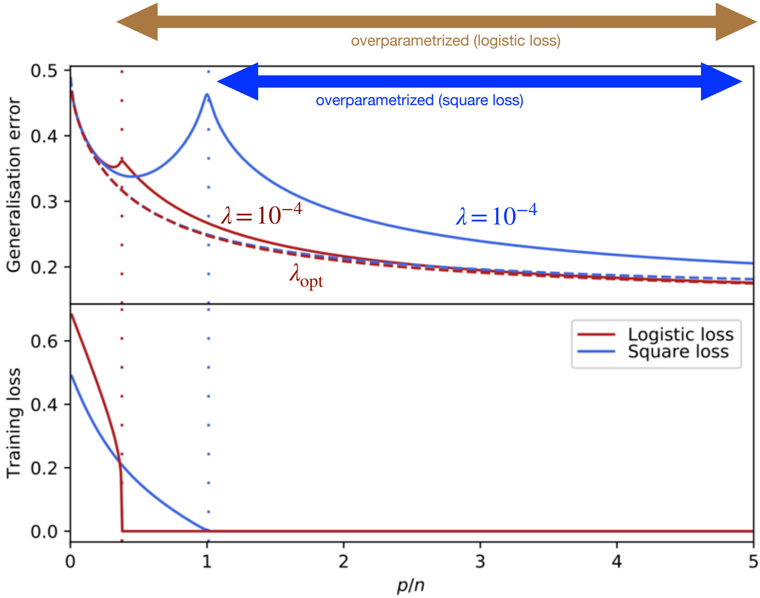
$$\mathcal{L}_{\text{training}} = \frac{\lambda}{2\alpha}q_w^\star + \mathbb{E}_{\xi,y}\left[\mathcal{Z}\left(y,\omega_0^\star\right)\ell\left(y,\eta(y,\omega_1^\star)\right)\right]$$

with $\omega_0^\star = M^\star/\sqrt{Q^\star}\xi, \omega_1^\star = \sqrt{Q^\star}\xi$

Agrees with **[Louart , Liao , Couillet'18 & Mei-Montanari '19]** who solved a particular case using random matrix theory: linear function $f^0$, $\ell(x,y) = \|x-y\|_2^2$ & Gaussian random weights **F**.

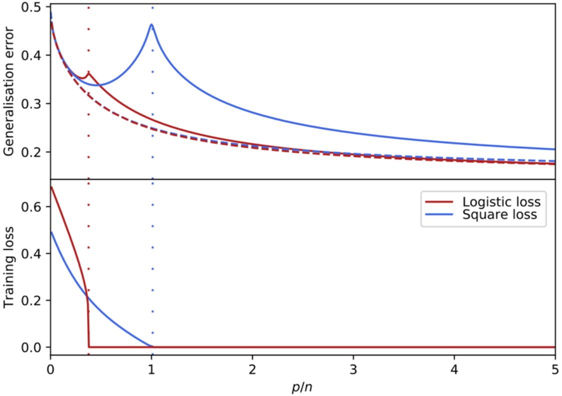
*slide by Florent Krzakala*

# A classification task




*slide by Florent Krzakala*

# A classification task




*Implicit regularisation of gradient descent* [Rosset, Zhy, Hastie, '04] [Neyshabur, Tomyoka, Srebro, '15]

*As $\lambda \to 0$, in the overparametrized regime,*
*Logistic converges to max-margin, $\ell_2$ converges to least norm*

slide by Florent Krzakala