From approximation theory to machine learning

New perspectives in the theory of function spaces and their applications

September 2017, Bedlewo, Poland

Jan Vybíral

Charles University/Czech Technical University
Prague, Czech Republic

Introduction

- Approximation Theory vs. Data Analysis (or Machine Learning, or Learning Theory)
- Example I: ℓ_1-minimization
- Example II: Support vector machines
- Example III: Ridge functions and neural networks

Approximation theory

- K - class of objects of interest: vectors, functions, operators,...
- ...approximated by (same or simpler) objects ...
- ...usually with only limited information about the unknown object...
- with the similarity (approximation error) measured by some sort of distance
- worst case - supremum of the approximation error over K
- average case - mean value of the (square of the) approximation error over K w.r.t. some probability measure on K

Approximation of multivariate functions

Let $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ be a function of many ($d \gg 1$) variables

We want to approximate f using only (a small number of) function values $f(x_1), \ldots, f(x_n)$

Questions:

- Which functions (assumptions on f)?
- How to measure the error?
- How to choose the sampling points?
- Decay of the error with growing number of sampling points?
- Algorithms and optimality?
- Dependence on d?

Data Analysis

Discovering structure in given data, allowing for predictions, conclusions, etc.

Given data can have different (and also heterogeneous) formats. For inputs $x_1, \ldots, x_n \in \mathbb{R}^d$ and outputs $y_1, \ldots, y_n \in \mathbb{R}$ we would like to study the functional dependence $y_i \approx f(x_i)$

The performance is in practice often tested by learning on a subset of the data and testing the rest.

To allow theoretical results, we often assume that the data is drawn independently from some (unknown) underlying probability distribution, i.e. $(x_i, y_i) \in A$ with prob. $\mathbb{P}(A)$.

Least squares & ridge regression

Least squares:

- $x_1, \ldots, x_n \in \mathbb{R}^d$: n data points (inputs) in $\mathbb{R}^d$
- $y_1, \ldots, y_n \in \mathbb{R}$: n real values (outputs)
- We look for dependence $y_i \approx \langle \omega, x_i \rangle$
- We minimize $\sum_{i=1}^{n} |y_i - \langle \omega, x_i \rangle|^2 = \|y - X\omega\|_2^2$ over $\omega \in \mathbb{R}^d$

Ridge regression: Adding regularization term $\lambda \|\omega\|_2^2$

LASSO: Adding regularization term $\lambda \|\omega\|_1$

Principal Component Analysis (PCA)

PCA: Classical dimensionality-reduction technique

- Given data points $x_1, \ldots, x_n \in \mathbb{R}^d$
- minimize over subspaces L with $\dim L = k$
- the distance of x_i's to L; i.e. $\min_{L:\dim L=k} \sum_{i=1}^{n} \|x_i - P_L x_i\|_2^2$
- the answer is given by singular value decomposition of the $n \times d$ data matrix.

"...swiss army knife of numerical linear algebra..."

Support Vector Machines

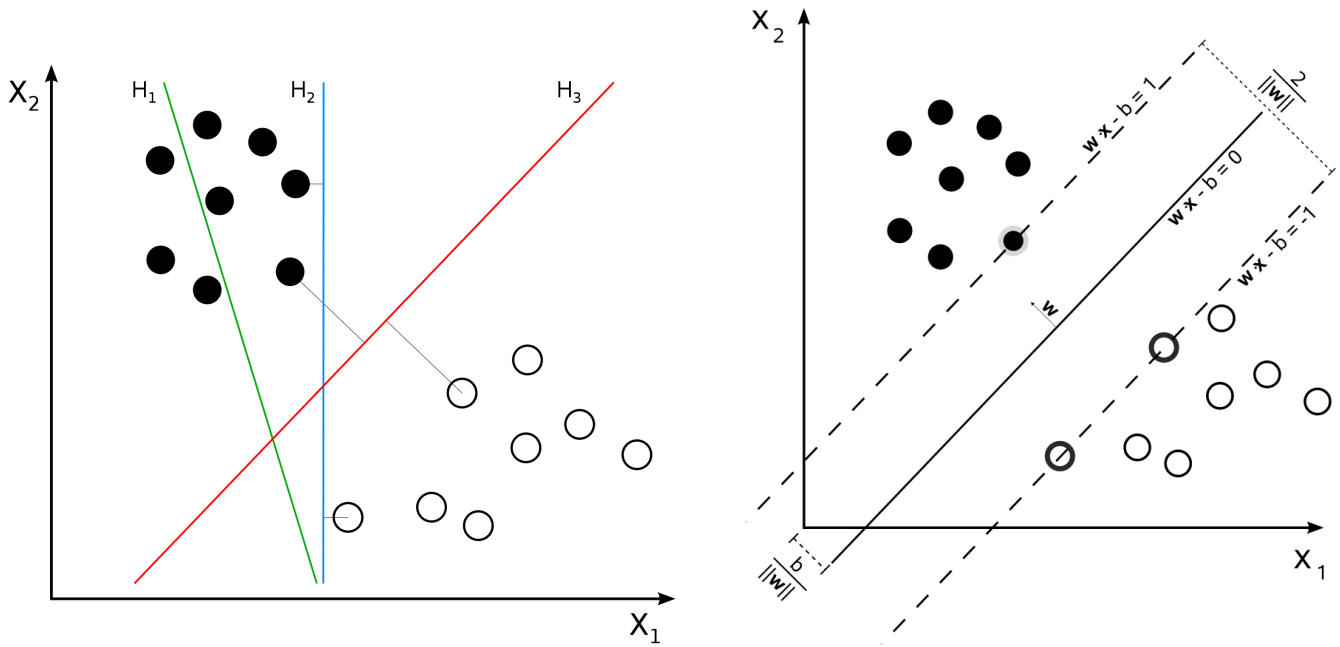
V. N. Vapnik and A. Ya. Chervonenkis (1963)
B. E. Boser, I. M. Guyon and V. N. Vapnik (1992) - non-linear
C. Cortes and V. N. Vapnik (1995) - soft margin

Given data points $x_1, \ldots, x_n \in \mathbb{R}^d$ and labels $y_i \in \{-1, 1\}$, separate the sets

$$\{x_i : y_i = -1\} \quad \text{and} \quad \{x_i : y_i = +1\}$$

by a linear hyperplane... i.e. $f(x) = g(\langle a, x \rangle)$, where

$$g(t) = \begin{cases} 1, & \text{if } t \geq 0, \\ -1, & \text{if } t < 0 \end{cases} \quad \text{is known.}$$



Soft margin SVM:

$$\min_{\omega \in \mathbb{R}^d} \sum_{i=1}^{n} (1 - y_i \langle \omega, x_i \rangle)_+ + \lambda \|\omega\|_2^2$$

Limits of the theory...

All theory, dear friend, is grey, but the golden tree of actual life springs ever green. *Johann Wolfgang von Goethe*

It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts. *Sherlock Holmes, A Scandal in Bohemia*

Approximation theory:

- The class of functions of interest is unclear.
- The data points sometimes can not be designed.

Data analysis - learning theory:

- The independence is not always ensured
- The underlying probability (and its basic properties) are unknown
- The success measured by leave-out data differs from one case to the other

Active choice of points!

link between data analysis and approximation theory

Difference between regression and sampling?

. . . active choice of the points . . .

Nowadays, more and more data analysis is done on calculated data!

In material science, material properties (like color or conductivity) are calculated *ab initio* from the molecular information, not measured in the lab.

Typically, these properties are governed by a PDE (like Schrödinger's equation), where the structure of the material comes in as initial data.

Calculation of one property for one material is then a numerical solution of a parametric: $\mathrm{PDE}(u, p, x) = 0$

The (unknown) functional dependence $f = f(p)$ of the material property on the input parameters is calculated for materials described with parameters $p_1, \ldots, p_m$

Sampling of f at $p_j \Leftrightarrow$
$\Leftrightarrow$ query of the solution $\Leftrightarrow$
$\Leftrightarrow$ running an expensive simulation

Nearly noise-free sampling possible!

Trade-off: more noisy samples vs. fewer less noisy samples

The materials with $p_1, \ldots, p_m$ can be actively chosen!

Summary of the introduction

- Approximation theory and Data Analysis often study similar problems from different points of view
- The choice of approximation algorithms is free in both areas
- Approximation theory (usually) assumes that the data can be designed/chosen
- The aim of the talk is to show the benefits of the combination of both approaches

ℓ_1-norm minimization

Data Analysis

LASSO (least absolute shrinkage and selection operator):

- Tibshirani (1996)
- Least squares with penalization term $\lambda\|\omega\|_1$
- $X = (x_1, \ldots, x_n) \in \mathbb{R}^{n\times d}$: n data points (inputs) in $\mathbb{R}^d$
- $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$: n real values (outputs)
- LASSO: $\arg\min_{\omega\in\mathbb{R}^d} \|y - X\omega\|_2^2 + \lambda\|\omega\|_1$ weights the fit $y \sim X\omega$ against the number of non-zero coordinates of ω
- Leaves open a number of questions - how many data points and how distributed are sufficient to identify/approximate ω...?

ℓ_1-norm minimization

Approximation theory

Compressed Sensing:

Let $w \in \mathbb{R}^d$ be sparse (with small number $s \ll d$ of non-zero coefficients on unknown positions) and consider the linear function $g(x) = \langle x, w \rangle$.

Design n (as small as possible) sampling points $x_1, \ldots, x_n$ and the function values $y_j = g(x_j) = \langle x_j, w \rangle$ and find a recovery mapping $\Delta : \mathbb{R}^n \to \mathbb{R}^d$, such that $\Delta(y)$ is equal/close to w

ℓ_1-norm minimization

Approximation theory

Compressed Sensing:

Let $w \in \mathbb{R}^d$ be sparse (with small number $s \ll d$ of non-zero coefficients on unknown positions) and consider the linear function $g(x) = \langle x, w \rangle$.

Design n (as small as possible) sampling points $x_1, \ldots, x_n$ and the function values $y_j = g(x_j) = \langle x_j, w \rangle$ and find a recovery mapping $\Delta : \mathbb{R}^n \to \mathbb{R}^d$, such that $\Delta(y)$ is equal/close to w

- Take x_j's independent at random, i.e. $y = Xw$
- Use ℓ_1-minimization for recovery

$$\Delta(y) = \underset{\omega \in \mathbb{R}^d}{\arg\min} \|\omega\|_1, \quad \text{s.t. } X\omega = y$$

- It is enough to take $n \approx s \log(d)$

ℓ_1-norm minimization

Approximation theory

Theorem (Donoho, Candes, Romberg, Tao - 2006)
There is a constant $C > 0$ such that if $m \geq Cs\log(d)$ and the components of x_i are independent Gaussian random variables, then $\Delta(y) = w$ (*with high probability*).

Many different aspects (other x_i's, noisy sampling, nearly-sparse vectors $w \in \mathbb{R}^d$, etc.) were studied intensively.

Benefits for Data Analysis:

- Estimates of the minimal number of data points
- If possible, choose mostly uncorrelated data points
- Better information about possibilities and limits of LASSO

ℓ_1-SVM

Data Analysis

ℓ_1-SVM:

- P.S. Bradley, O.L. Mangasarian (1998)
  J. Zhu, S. Rosset, T. Hastie, R. Tibshirani (2004)
- Support vector machine with ℓ_1-penalization term
- $X = (x_1, \ldots, x_n) \in \mathbb{R}^{n \times d}$: n data points (inputs) in $\mathbb{R}^d$
- $y = (y_1, \ldots, y_n) \in \{-1, +1\}$: n labels
- ℓ_1-SVM:

$$\min_{\omega \in \mathbb{R}^d} \sum_{i=1}^{n} (1 - y_i \langle \omega, x_i \rangle)_+ + \lambda \|\omega\|_1$$

- Weights the quality of the classification against the number of non-zero coordinates of ω
- Leaves open a number of questions - how many data points and how distributed are sufficient to identify ω...?

ℓ_1-SVM

Data Analysis

- Standard technique of sparse classification
- Bioinformatics
  - gene selection
  - microarray classification
  - cancer classification
- feature selection
- face recognition
- ...

ℓ_1-SVM

Data Analysis

Results:

I. Steinwart, Support vector machines are universally consistent, J. Complexity (2002)

I. Steinwart, Consistency of support vector machines and other regularized kernel classifiers, IEEE Trans. Inf. Theory (2005):

The SVM's are consistent (in general setting with kernels), i.e. they can learn the hidden classifier for $n \to \infty$ almost surely; the parameter of SVM has to grow to infinity.

ℓ_1-SVM

Approximation theory

- We want to identify sparse (or nearly sparse) $\omega \in \mathbb{R}^d$
- We assume $\omega \in K = \{x \in \mathbb{R}^d : \|x\|_p \leq 1\}, p < 1$
- We are allowed to take only 1-bit measurements $y_i = \text{sign}(\langle \omega, x_i \rangle)$ - non-linear!
- We are allowed to design the "sampling points" $x_1, \ldots, x_n \in \mathbb{R}^d$ and a (non-linear) recovery algorithm
- Recent area of **1-bit Compressed Sensing**

ℓ_1-SVM

Approximation theory

1-bit Compressed Sensing:
Boufounos & Baraniuk (2008), Y. Plan & R. Vershynin (2013)

$$\omega \in K = \{x \in \mathbb{R}^d : \|x\|_2 \leq 1,\ \|x\|_1 \leq \sqrt{s}\}$$
$$x_i \sim \mathcal{N}(0, \mathrm{id}), \quad y_i = \mathrm{sign}(\langle \omega, x_i \rangle)$$
$$\hat{\omega} := \arg\max_{w \in K} \sum_{i=1}^{n} y_i \langle x_i, w \rangle$$

Result: For $n \geq C\delta^{-6} w(K)^2$

$$\|\hat{\omega} - \omega\|_2^2 \leq \delta \sqrt{\log(e/\delta)} \text{ with prob. } \geq 1 - 8\exp(-c\delta^2 n)$$

Mean Gaussian width: $w(K) = \mathbb{E} \sup_{x \in K-K} \langle g, x \rangle$

ℓ_1-SVM

Approximation theory

What if we insist on ℓ_1-SVM as the recovery algorithm?

Setting:

- $\|\omega\|_2 = 1, \quad \|\omega\|_1 \leq R$
- $x_i \sim \mathcal{N}(0, r^2 \cdot \mathrm{id}), \quad i = 1, \ldots, n$
- $y_i = \mathrm{sign}(\langle x_i, \omega \rangle), \quad i = 1, \ldots, n$
- $\hat{\omega} \in \mathbb{R}^d$ a minimizer of the ℓ_1-SVM

$$\min_{w \in \mathbb{R}^d} \sum_{i=1}^{n} [1 - y_i \langle x_i, w \rangle]_+ \quad \text{subject to} \quad \|w\|_1 \leq R.$$

Theorem (Kolleck, V. (2016))

Let $0 < \varepsilon < 0.18$, $r > \sqrt{2\pi}(0.57 - \pi\varepsilon)^{-1}$, $n \geq C\varepsilon^{-2} r^2 R^2 \ln(d)$

Then it holds

$$\left\| \omega - \frac{\hat{\omega}}{\|\hat{\omega}\|_2} \right\|_2 \leq C' \left(\varepsilon + \frac{1}{r} \right)$$

with probability at least $1 - \exp\left(-C'' \ln(d)\right)$

ℓ_1-SVM

Summary on SVM's:

- Non-asymptotic analysis of (ℓ_1-)SVM's allows to predict the limits of the algorithm
- For random (i.e. highly uncorrelated data) small number n of measurements/data is sufficient
- Motivated by the analysis, we proposed an SVM, which combines the ℓ_1 and the ℓ_2 penalty; this method performs better in analysis, but also in the numerical simulation!
- Use of the 1-Bit CS algorithm in Machine Learning?

Ridge functions & Neural networks

Ridge functions

Let $g : \mathbb{R} \to \mathbb{R}$ and $a \in \mathbb{R}^d \setminus \{0\}$.

Ridge function with *ridge profile* g and *ridge vector* a is the function

$$f(x) := g(\langle a, x \rangle).$$

Constant along the hyperplane $a^\perp = \{y \in \mathbb{R}^d : \langle y, a \rangle = 0\}$ and its translates.

More general, if $g : \mathbb{R}^k \to \mathbb{R}$ and $A \in \mathbb{R}^{k \times d}$ with $k \ll d$ then

$$f(x) := g(Ax)$$

is a $k-$*ridge function*.

Ridge functions in mathematics

- Kernels of important transforms (Fourier, Radon)
- Plane waves in PDE's:
  Solutions to
  $$\prod_{i=1}^{r}\Big(b_i\frac{\partial}{\partial x}-a_i\frac{\partial}{\partial y}\Big)F=0$$
  are of the form
  $$F(x,y)=\sum_{i=1}^{r}f_i(a_ix+b_iy).$$
- Ridgelets, curvelets, shearlets,. . . : wavelet-like frames capturing singularities along curves and edges (Candes, Donoho, Kutyniok, Labate, . . .)

Ridge functions in approximation theory

Approximation of a function by functions from the dictionary

$$D_{\text{ridge}} = \{\varrho(\langle k, x\rangle - b) : k \in \mathbb{R}^d, b \in \mathbb{R}\}$$

- Fundamentality
- Greedy algorithms

Lin & Pinkus, Fundamentality of ridge functions, J. Approx. Theory 75 (1993), no. 3, 295–311

Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Systems 2 (1989), 303–314

Leshno, Lin, Pinkus & Schocken, Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, Neural Networks 6 (1993), 861–867

Neural networks

Motivated by biological research on human brain and neurons
W. McCulloch, W. Pitts (1943); M. Minsky, S. Papert (1969)
Artificial Neuron:
. . . gets activated if a linear combination of its inputs grows over a certain threshold. . .

- Inputs $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$
- Weights $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$
- Comparing $\langle w, x \rangle$ with a threshold $b \in \mathbb{R}$
- Plugging the result into the "activation function" - jump (or smoothed jump) function σ

Artificial neuron is a function

$$x \to \sigma(\langle x, w \rangle - b),$$

where $\sigma : \mathbb{R} \to \mathbb{R}$ might be $\sigma(x) = \text{sign}(x)$ or $\sigma(x) = e^x/(1 + e^x)$, etc.

Neural networks

Artificial neural network is a directed, acyclic graph of artificial neurons

- Input: $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$
- First layer of neurons:
  $y_1 = \sigma(\langle x, w_1^1 \rangle - b_1^1), \ldots, y_{n_1} = \sigma(\langle x, w_{n_1}^1 \rangle - b_{n_1}^1)$
- The outputs $y = (y_1, \ldots, y_{n_1})$ become inputs for the next layer $\ldots$; last layer outputs $y \in \mathbb{R}$
- "Deep Learning" relies on an artificial neural network with $\sim 100 - 1000$ layers
- Training the network: given inputs $x^1, \ldots, x^N$ and outputs $y^1, \ldots, y^N$ and optimize over weights w's and b's
- Non-convex minimization over a huge space...???

Approximation of ridge functions (and their sums)

$k = 1$: $f(x) = g(\langle a, x \rangle), \quad \|a\|_2 = 1, \quad g$ smooth

Approximation has two parts: approximation of g and of a

Recovery of a - from $\nabla f(x)$:

$$\nabla f(x) = g'(\langle a, x \rangle)a, \nabla f(0) = g'(0)a.$$

After recovering a, the problem becomes essentially one-dimensional and one can use arbitrary sampling method to approximate g.

$g'(0) \neq 0 \ldots g'(0) = 1$

Buhmann & Pinkus '99
Identifying linear combinations of ridge functions

Approximation of functions

$$f(x) = \sum_{i=1}^{m} g_i(\langle a_i, x \rangle), \quad x \in \mathbb{R}^d$$

- $g_i \in C^{2m-1}(\mathbb{R})$, $i = 1, \ldots, m$;
- $g_i^{(2m-1)}(0) \neq 0$, $i = 1, \ldots, m$;

$$(D_u^{2m-1-k} D_v^k f)(0) = \sum_{i=1}^{m} (\langle u, a_i \rangle)^{2m-1-k} (\langle v, a_i \rangle)^k g_i^{(2m-1)}(0)$$

for $k = 0, \ldots, 2m-1$ and $v_1, \ldots, v_d \in \mathbb{R}^d$ and solving this system of equations.

A.Cohen, I.Daubechies, R.DeVore, G.Kerkyacharian, D.Picard, '12
Capturing ridge functions in high dimensions from point queries

- $\boldsymbol{k = 1 : f(x) = g(\langle a, x \rangle)}$
- $f : [0,1]^d \to \mathbb{R}$
- $g \in C^s([0,1])$, $1 < s$
- $\|g\|_{C^s} \leq M_0$
- $\boldsymbol{\|a\|_{\ell_q^d} \leq M_1, 0 < q \leq 1}$
- $\boldsymbol{a \geq 0}$

Then

$$\|f - \hat{f}\|_\infty \leq C M_0 \left\{ L^{-s} + M_1 \left(\frac{1 + \log(d/L)}{L} \right)^{1/q-1} \right\}$$

using $3L + 2$ sampling points

- First sampling along the diagonal $\frac{i}{L}\mathbf{1} = \frac{i}{L}(1,\dots,1), i = 0,\dots,L$:

$$f\left(\frac{i}{L}\mathbf{1}\right) = g\left(\left\langle \frac{i}{L}\mathbf{1}, a\right\rangle\right) = g(i\|a\|_1/L)$$

- Recovery of g on a grid of $[0, \|a\|_1]$
- Finding i_0 with largest $g((i_0+1)\|a\|_1/L) - g(i_0\|a\|_1/L)$
- Approximating $D_{\varphi_j} f(i_0/L \cdot \mathbf{1}) = g'(i_0\|a\|_1/L)\langle a, \varphi_j\rangle$ by first order differences
- Then recovery of a from $\langle a, \varphi_1\rangle, \dots, \langle a, \varphi_m\rangle$ by methods of compressed sensing (CS)

M. Fornasier, K. Schnass, J. V., *Learning functions of few arbitrary linear parameters in high dimensions* (2012)

- $f : B(0,1) \to \mathbb{R}$
- $\|a\|_2 = 1$
- $0 < q \leq 1, \|a\|_q \leq c$
- $g \in C^2[-1,1]$

Put

$$y_j := \frac{f(h\varphi_j) - f(0)}{h} \approx g'(0)\langle a, \varphi_j \rangle, \quad j = 1, \dots, m_\Phi, \quad m_\Phi \leq d,$$

where $h > 0$ is small, and

$$\varphi_{j,k} = \pm \frac{1}{\sqrt{m_\Phi}}, \quad k = 1, \dots, d$$

y_j are scalar products $\langle a, \varphi_j \rangle$ corrupted by deterministic noise

$$\tilde{a} = \arg\min_{z \in \mathbb{R}^d} \|z\|_1, \quad \text{s.t. } \langle \varphi_j, z \rangle = y_j, \quad j = 1, \dots, m_\Phi.$$

$\hat{a} = \tilde{a}/\|\tilde{a}\|_2$ - approximation of a

$\hat{g}$ is obtained by sampling f along $\hat{a}$: $\hat{g}(y) := f(\hat{a} \cdot t), \ t \in (-1, 1)$.

Then

$$\hat{f}(x) := \hat{g}(\langle \hat{a}, x \rangle),$$

has the approximation property

$$\|f - \hat{f}\|_\infty \le C \Big[\left(\frac{m_\Phi}{\log(d/m_\Phi) + 1} \right)^{-\left(\frac{1}{q} - \frac{1}{2}\right)} + \frac{h}{\sqrt{m_\Phi}} \Big].$$

Active coordinates:

R. DeVore, G. Petrova, P. Wojtaszczyk, *Approximation of functions of few variables in high dimensions*, Constr. Appr. '11

K. Schnass, J.V., Compressed learning of high-dimensional sparse functions, Proceedings of ICASSP '11

$$f(x) = g(x_{i_1}, \ldots, x_{i_k})$$

Use of low-rank matrix recovery:

H. Tyagi, V. Cevher, *Learning non-parametric basis independent models from point queries via low-rank methods*, ACHA '14

I. Daubechies, M. Fornasier, J.V., *Approximation of sums of ridge functions*, in preparation: **Sums of ridge functions**

Recovery of

$$f(x) = \sum_{j=1}^{k} g_j(\langle a_j, x \rangle)$$

- We would like to identify $a_1, \ldots, a_k$, then $g_1, \ldots, g_k$
- **Step 1.:** Sampling of

$$\nabla f(x) = \sum_{j=1}^{k} g_j'(\langle a_j, x \rangle) a_j$$

  at different points gives elements of $\mathrm{span}\{a_1, \ldots, a_k\} \subset \mathbb{R}^d$
- Afterwards, we can reduce the dimension to $d = k$

  ...one k-dimensional problem...

Recovery of individual a_i's for $d = k$?

- **Step 2.:** Second order derivatives:

$$\nabla^2 f(x) = \sum_{j=1}^{k} g_j''(\langle a_j, x \rangle) a_j \otimes a_j$$

- We can recover $\tilde{L}$ - an approximation of

$$L = \text{span}\{a_i \otimes a_i, i = 1, \ldots, k\} \subset \mathbb{R}^{k \times k}$$

- **Step 3.:** We try to find $a_i \otimes a_i$ in $\tilde{L}$
- We look for matrices in $\tilde{L}$ with the smallest rank
- We analyze the non-convex problem

$$\arg\max \|M\|, \quad \text{s.t.} \quad M \in \tilde{L}, \|M\|_F \leq 1$$

- Every algorithm, which is able to find $a_1 \otimes a_1$ can also find $a_j \otimes a_j$, $j = 2, \ldots, k$, hence **it must be non-convex**

Summary

- Approximation theory and Machine Learning study similar problems from different points of view
- They can inspire/enrich one the other
- Non-linear, non-convex classes; non-linear information
- Sparsity and other structural assumptions
- **Open problem:** Geometrical properties of the class of functions, which can be modeled by neural networks

$$\mathcal{N}_{d,L} = \{f : \mathbb{R}^d \to \mathbb{R} : f \text{ is a neural network with } L \text{ layers}\}$$

Thank you for your attention!