Workshop "Apprentissage et simulation en grande dimension",
Airbus Group, June 24-26, 2019

Deep tensor networks

Introduction

Anthony Nouy

Centrale Nantes, Laboratoire de Mathématiques Jean Leray

High-dimensional problems

Many problems of computational science, probability and statistics require the approximation, integration or optimization of functions of many variables

$$u(x_1, \ldots, x_d)$$

High-dimensional problems in mechanics and physics

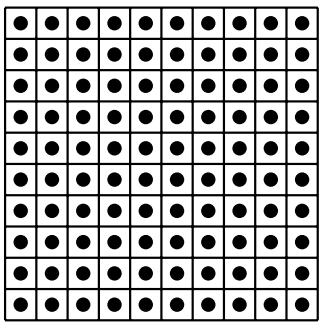
- Navier Stokes equation

$$u(x,t)$$

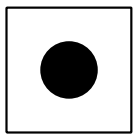
$$\frac{\partial u}{\partial t} + u \cdot \nabla u - \nu \Delta u = f$$

- Multiscale problems

$$u(x,y,t), \quad x \in \Omega,\ y \in Y$$



Ω



Y

High-dimensional problems in mechanics and physics

- Boltzmann equation

$$f(x, p, t)$$

$$\frac{\partial f}{\partial t} + m^{-1} p \cdot \frac{\partial f}{\partial x} + F \cdot \frac{\partial f}{\partial p} = g$$

- Fokker-Planck equation

$$p(x_1, \ldots, x_d, t)$$

$$\frac{\partial p}{\partial t} + \sum_{i=1}^{d} \frac{\partial}{\partial x_i}(a_i p) - \frac{1}{2} \sum_{i,j=1}^{d} \frac{\partial^2}{\partial x_i x_j}(b_{ij} p) = 0$$

- Schrödinger equation

$$\Psi(x_1, \ldots, x_d, t)$$

$$i\hbar \frac{\partial \Psi}{\partial t} = -\frac{\hbar}{2\mu} \Delta \Psi + V \Psi$$

High-dimensional problems in statistics and data science

- Unsupervised learning. Estimation of the probability distribution

$$F(x_1, \ldots, x_d) = \mathbb{P}(X_1 \leq x_1, \ldots, X_d \leq x_d),$$

  of a random vector $X = (X_1, \ldots, X_d)$, from samples of X or some function of X.

- Supervised learning. Approximation of a random variable Y by a function of a set of random variables $X = (X_1, \ldots, X_d)$, using samples of (X, Y). The approximation is used as a predictive model.

- These are two typical tasks in uncertainty quantification, where Y is some output variable of a (numerical or experimental) model depending on a set of random parameters X.

Outline

Outline

Approximation

The goal is to approximate a function

$$u(x_1, \ldots, x_d)$$

by an element of a subset of functions X_n described by n parameters.

- X_n is called an approximation tool, model class or hypothesis set.
- Standard approximation tools include splines, wavelets, polynomials, with or without adaptivity.
- We distinguish linear approximation, where X_n are linear spaces, from nonlinear approximation, where X_n are nonlinear spaces.

Approximation

For a function u from a normed space, the best approximation error

$$e_n(u) = \inf_{v \in X_n} \|u - v\|,$$

quantifies what we can expect from X_n.

Fundamental problems are to

- determine the complexity $n = n(\epsilon, u)$ required for obtaining an error

$$e_n(u) \leq \epsilon,$$

- provide algorithms that practically compute approximations achieving this precision with almost optimal complexity, using available information on the function (model equations, samples,...)

The curse of dimensionality

For a function u from classical regularity classes (Sobolev or Besov spaces), it is known that

$$n(\epsilon, u) \lesssim \epsilon^{-d/k}$$

for standard approximation tools (splines, wavelets).

- We observe that $n(\epsilon, u)$ grows exponentially with the dimension d, which is the curse of dimensionality.
- A better performance may be observed for particular functions and particular approximation tools.
- But a priori, we can not expect a better performance from any (reasonable) approximation tool without further assumptions on the function.

How to beat the curse of dimensionality ?

We have to

- make assumptions on the structure of the function, going ahead standard regularity assumptions,
- propose approximation tools (model classes) that capture these structures.

Some standard model classes

- Linear models

$$a_1 x_1 + \ldots + a_d x_d$$

- Polynomial models

$$\sum_{\alpha \in \Lambda} a_\alpha x_1^{\alpha_1} \ldots x_d^{\alpha_d}$$

or more general sparse tensors

$$\sum_{\alpha \in \Lambda} a_\alpha \varphi^1_{\alpha_1}(x_1) \ldots \varphi^d_{\alpha_d}(x_d)$$

where $\Lambda \subset \mathbb{N}^d$ is a set of multi-indices, either fixed (linear approximation) or free (nonlinear approximation).

Curse of dimensionality can be circumvented for functions with sufficient anisotropy

Some standard model classes

- Additive models

$$u_1(x_1) + \ldots + u_d(x_d)$$

or more generally

$$\sum_{\alpha \subset T} u_\alpha(x_\alpha)$$

where $T \subset 2^{\{1,\ldots,d\}}$ is either fixed (linear approximation) or a free parameter (nonlinear approximation).

- Multiplicative models

$$u_1(x_1) \ldots u_d(x_d)$$

or more generally

$$\prod_{\alpha \in T} u_\alpha(x_\alpha)$$

where $T \subset 2^{\{1,\ldots,d\}}$ is either a fixed or a free parameter. An instance of graphical models.

Composition of functions

$$f(g(x))$$

using standard model classes for both f and g.

- Linear transformations (ridge functions)

$$f(Wx), \quad W \in \mathbb{R}^{m \times d}$$

  - With an additive model for f, projection pursuit

$$f_1(w_1^T x) + \ldots + f_m(w_m^T x)$$

  - A more specific case is the sum of m perceptrons (shallow neural network with one hidden layer of width m)

$$\sum_{i=1}^{m} a_i \sigma(w_i^T x + b_i)$$

- Sparse transformations, e.g.

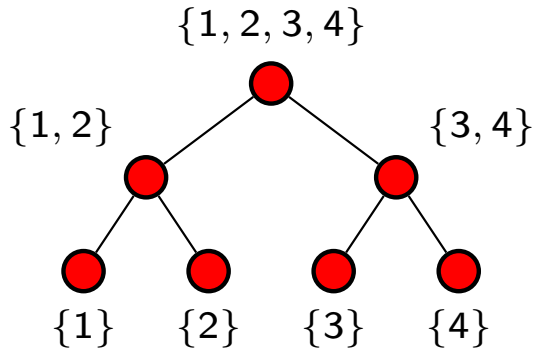
$$f(g_{1,2}(x_1, x_2), g_{3,4}(x_3, x_4), \ldots)$$

More compositions... deep neural networks

$$f \circ g_L \circ g_{L-1} \circ \ldots \circ g_2 \circ g_1(x)$$

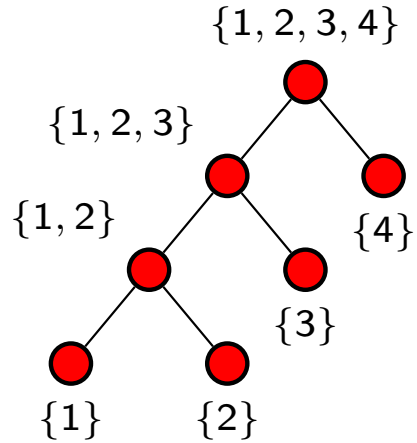
- Convolutional networks, sparse transformations with sparsity induced by a balanced tree

$$f_{1,2,3,4}\left(f_{1,2}\left(f_1(x_1), f_2(x_2)\right), f_{3,4}\left(f_3(x_3), f_4(x_4)\right)\right)$$



- Reccurent networks, sparse transformations with sparsity induced by a linear tree

$$f_{1,2,3,4}\left(f_{1,2,3}\left(f_{1,2}\left(f_1(x_1), f_2(x_2)\right), f_3(x_3)\right), f_4(x_4)\right)$$



More compositions... deep neural networks

These are highly nonlinear approximation tools, with a high approximation power.

They are known to achieve the optimal performance for standard regularity classes, but we can not expect better than classical tools without further assumptions on the function.

Even if the expected error $e_n(u)$ is small for a certain function u,

- there is no known certified algorithm for constructing an approximation achieving this error,
- and a best approximation (when it exists) may be highly unstable.

Outline

Low-rank formats

- Functions with rank one (multiplicative model)

$$v(x) = u_1(x_1) \dots u_d(x_d)$$

- Functions with canonical rank less than r (canonical format)

$$v(x) = \sum_{i=1}^{r} u_1^i(x_1) \dots u_d^i(x_d)$$

Low rank formats

- For a subset of variables $\alpha \subset \{1, \ldots, d\} := D$, $v(x)$ can be identified with a bivariate function

$$v(x_\alpha, x_{\alpha^c}),$$

where x_α and x_{α^c} are complementary groups of variables.

The canonical rank of this bivariate function is called the α-rank of v, denoted $\text{rank}_\alpha(v)$, which is the minimal integer r_α such that

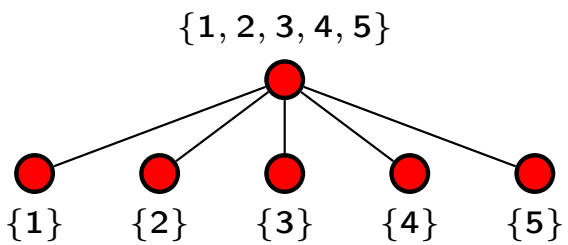
$$v(x) = \sum_{k=1}^{r_\alpha} v_k^\alpha(x_\alpha) w_k^{\alpha^c}(x_{\alpha^c})$$

Tree based tensor formats

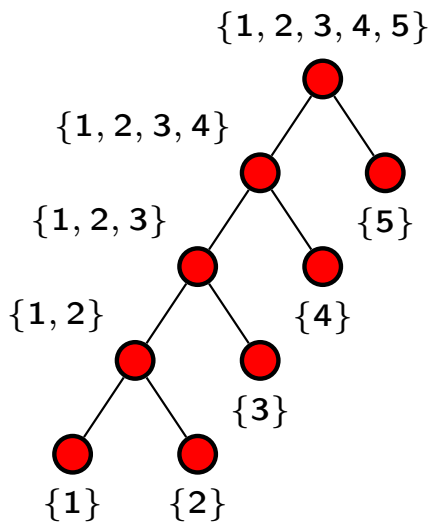
- For $T \subset 2^D$ a collection of subsets of D, a tensor format is defined by

$$\mathcal{T}_r^T = \{v : \text{rank}_\alpha(v) \leq r_\alpha, \alpha \in T\}.$$

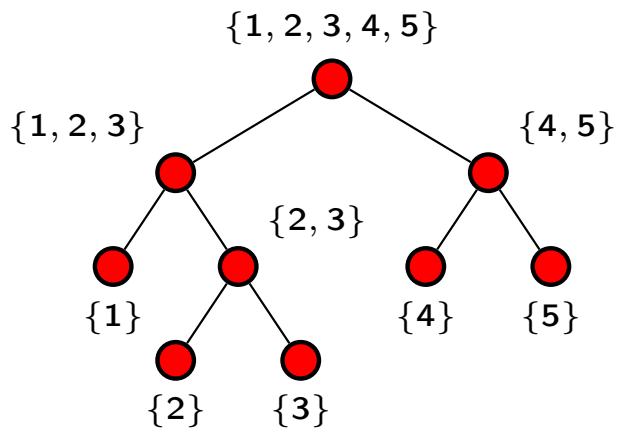
- In the particular case where T is a dimension partition tree, $\mathcal{T}_r^T$ is a tree-based tensor format.



Tucker



Tensor Train



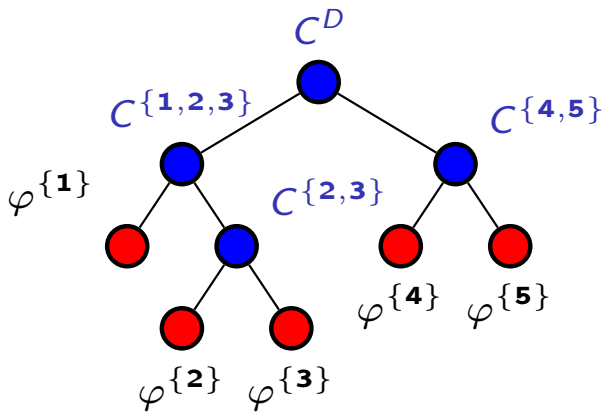
Hierarchical Tucker

Tree-based tensor formats as deep tensor networks

- A tensor v in $\mathcal{T}_r^T$ admits a multilinear parametrization

$$v(x) = \sum_{\substack{1 \le k_\alpha \le r_\alpha \\ \alpha \in T}} \prod_{\gamma \in T \setminus \mathcal{L}(T)} C^\gamma_{k_\gamma, (k_\beta)_{\beta \in S(\gamma)}} \prod_{\gamma \in \mathcal{L}(T)} \varphi^\gamma_{k_\gamma}(x_\gamma)$$

with parameters C^γ and φ^γ forming a tree network of low order tensors.

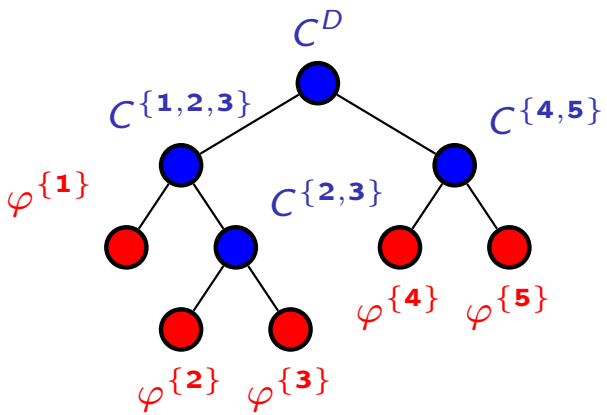


- Storage complexity scales as $O(dR^{s+1} + dNR)$ where R is the maximal α-rank, s is the arity of the tree, and N is the storage complexity of a function $\varphi^\gamma_{k_\gamma}$.

Tree-based tensor formats as compositions of multilinear functions

For each node α with children $\{\beta_1, \ldots, \beta_s\}$, the tensor C^α in $\mathbb{R}^{r_{\beta_1} \times \ldots \times r_{\beta_s} \times r_\alpha}$ can be identified with a multilinear map from $\mathbb{R}^{r_{\beta_1}} \times \ldots \times \mathbb{R}^{r_{\beta_s}}$ to $\mathbb{R}^{r_\alpha}$. Then tree-based format can be written as a composition of multilinear functions.

For example,



$$v(x) = C^D\left(C^{\{1,2,3\}}\left(\varphi^{\{1\}}(x_1), C^{\{2,3\}}\left(\varphi^{\{2\}}(x_2), \varphi^{\{3\}}(x_3)\right)\right), C^{\{4,5\}}\left(\varphi^{\{4\}}(x_4), \varphi^{\{5\}}(x_5)\right)\right)$$

The format corresponds to a deep neural network with a sparse architecture given by the tree and multilinear functions.

Computing with tree-based tensor formats

Many favorable properties from a computational point of view.

- Complexity is linear in d and polynomial in the rank for storage, evaluation, differentiation, integration...
- Not so nonlinear approximation tool
- Topological properties ensure the well-posedness of optimization problems and existence of stable algorithms
- Geometrical properties can be exploited for optimization and dynamical approximation.
- Notion of higher-order singular value decomposition

$$u(x) = \sum_{k \geq 1} \sigma_k^{\alpha} v_k(x_\alpha) w_k(x_{\alpha^c})$$

and a way to obtain approximations u_r in $\mathcal{T}_r^T$ such that

$$\|u - u_r\| \leq \sqrt{2d} \inf_{v \in \mathcal{T}_r^T} \|u - v\|.$$

Approximation properties of tree-based tensor formats

- For standard regularity classes (e.g. Sobolev spaces), they perform almost as well as standard approximation tools (splines, wavelets)...
- but they can perform much better for non standard classes of functions, e.g. compositions of smooth functions

$$f_{1,2,3,4}\left(f_{1,2}\left(f_1(x_1), f_2(x_2)\right), f_{3,4}\left(f_3(x_3), f_4(x_4)\right)\right)$$

- A function in canonical format (shallow tensor network)

$$u(x) = \sum_{k=1}^{r} u_k^1(x_1) \dots u_k^d(x_d)$$

  can be represented in tree-based format with a similar complexity.
- Conversely, a typical function in tree-based format $\mathcal{T}_r^T$ has a canonical rank depending exponentially in the dimension d.

Deep is better !

Approximation properties of tree-based tensor formats

As an example, consider the probability distribution $f(x) = \mathbb{P}(X = x)$ of a Markov chain $X = (X_1, \ldots, X_d)$ given by

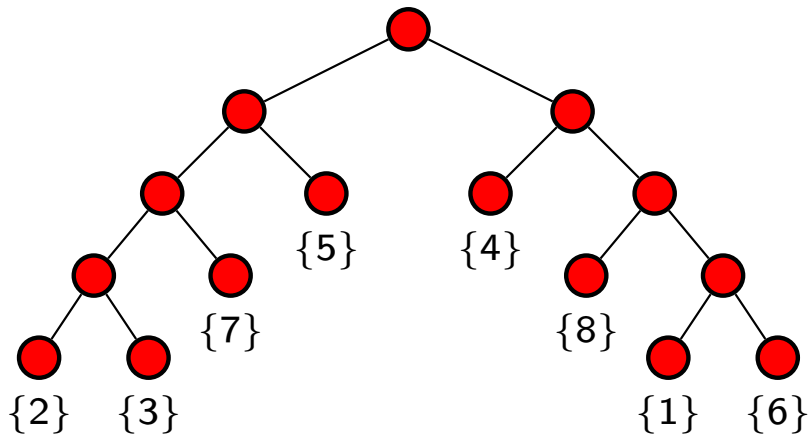
$$f(x) = f_1(x_1) f_{2|1}(x_2|x_1) \ldots f_{d|d-1}(x_d|x_{d-1})$$

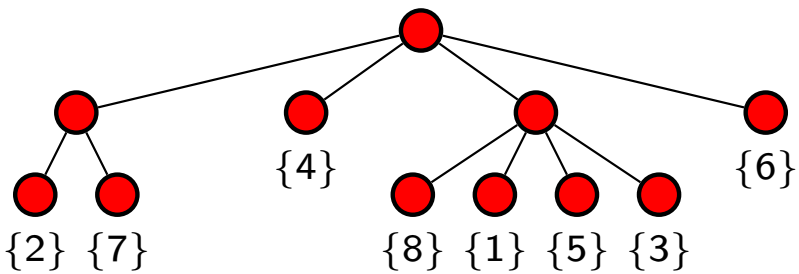
where bivariate functions $f_{i|i-1}$ have a rank bounded by r.

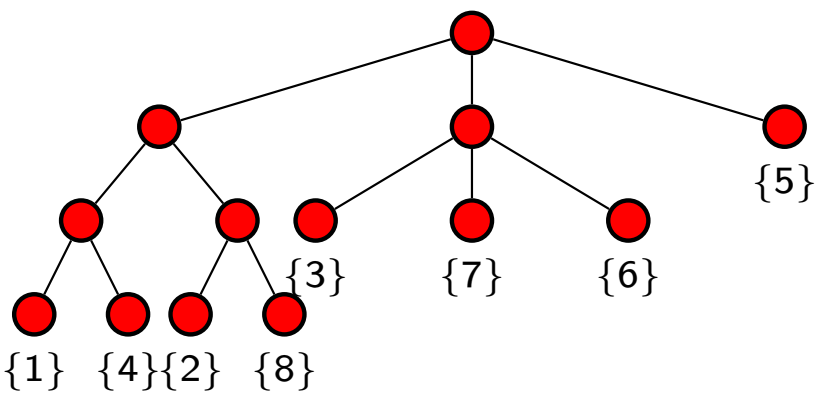
- With the linear tree T containing interior nodes $\{1,2\}, \{1,2,3\}, \ldots, \{1, \ldots, d-1\}$, f admits a representation in tree-based format with storage complexity in r^4.
- The canonical rank of f is exponential in d.
- But when considering the linear tree T_σ obtained by applying permutation $\sigma = (1, 3, \ldots, d-1, 2, 4, \ldots, d)$ to the tree T, the storage complexity in tree-based format is also exponential in d.

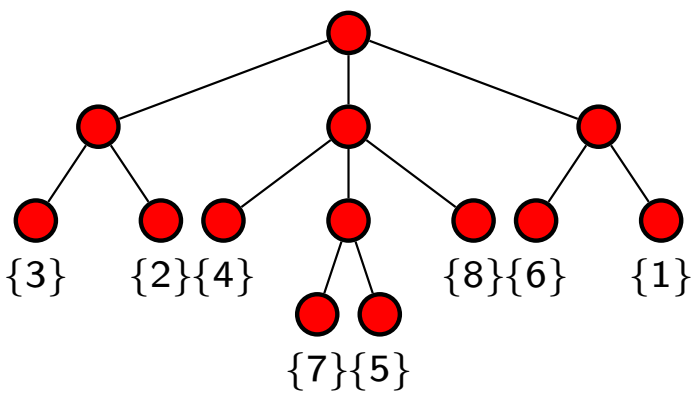
Approximation properties of tree-based tensor formats

Choosing a good tree (architecture of network) is a crucial but combinatorial issue...









Outline

Supervised and unsupervised learning

Two typical tasks of learning are to

- estimate the probability distribution of a random vector $Z = (Z_1, \ldots, Z_d)$ from samples of the distribution (unsupervised learning)
- approximate a random variable Y by a function of a set of variables $X = (X_1, \ldots, X_d)$, from samples of the pair $Z = (X, Y)$ (supervised learning)

Risk

A classical approach is to introduce a constrast (or loss) function $\gamma(v, z)$ and associated risk (expected loss)

$$\mathcal{R}(v) = \mathbb{E}(\gamma(v, Z))$$

whose minimizer over the set of functions v is the target function u (or oracle) and such that

$$\mathcal{R}(v) - \mathcal{R}(u)$$

measures some distance between the target u and the function v.

- For least-squares regression in supervised learning, $\mathcal{R}(v) = \mathbb{E}((Y - v(X))^2$, and $\mathcal{R}(v) - \mathcal{R}(u) = \mathbb{E}((u(X) - v(X))^2)$.
- For unsupervised learning with L^2-loss, $\mathcal{R}(v) = \mathbb{E}(\|v\|^2 - 2v(Z))$ and $\mathcal{R}(v) - \mathcal{R}(u)$ is the L^2 distance between v and the probability density of Z.

Empirical risk minimization

Given i.i.d. samples $\{z_i\}_{i=1}^N$ of Z, an approximation $\hat{u}_F^N$ of u is obtained by minimization of the empirical risk

$$\widehat{\mathcal{R}}_n(v) = \frac{1}{N}\sum_{i=1}^{N} \gamma(v, z_i)$$

over a certain model class F.

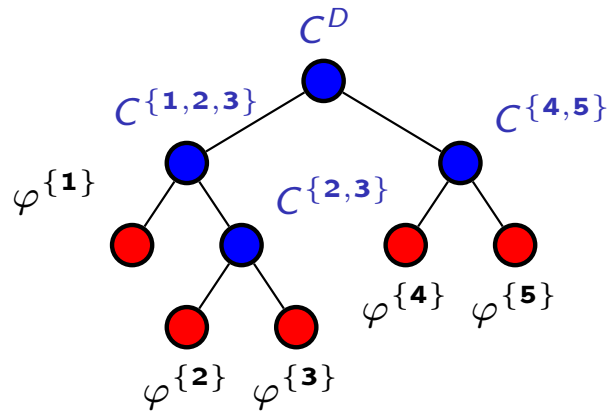
- Denoting by u_F the minimizer of the risk over F, the error

$$\mathcal{R}(\hat{u}_F^N) - \mathcal{R}(u) = \underbrace{\mathcal{R}(\hat{u}_F^N) - \mathcal{R}(u_F)}_{\text{estimation error}} + \underbrace{\mathcal{R}(u_F) - \mathcal{R}(u)}_{\text{approximation error}}$$

- For a given sample, when taking larger and larger model classes, approximation error $\searrow$ while estimation error $\nearrow$.
- Adaptive methods should be proposed for the selection of a model class taking the best from the available information.

Learning with tree tensor networks

$$v(x) = \sum_{\substack{1 \leq k_\alpha \leq r_\alpha \\ \alpha \in \overline{T}}} \prod_{\gamma \in T \setminus \mathcal{L}(T)} C^{\gamma}_{k_\gamma, (k_\beta)_{\beta \in S(\gamma)}} \prod_{\gamma \in \mathcal{L}(T)} \varphi^{\gamma}_{k_\gamma}(x_\gamma)$$



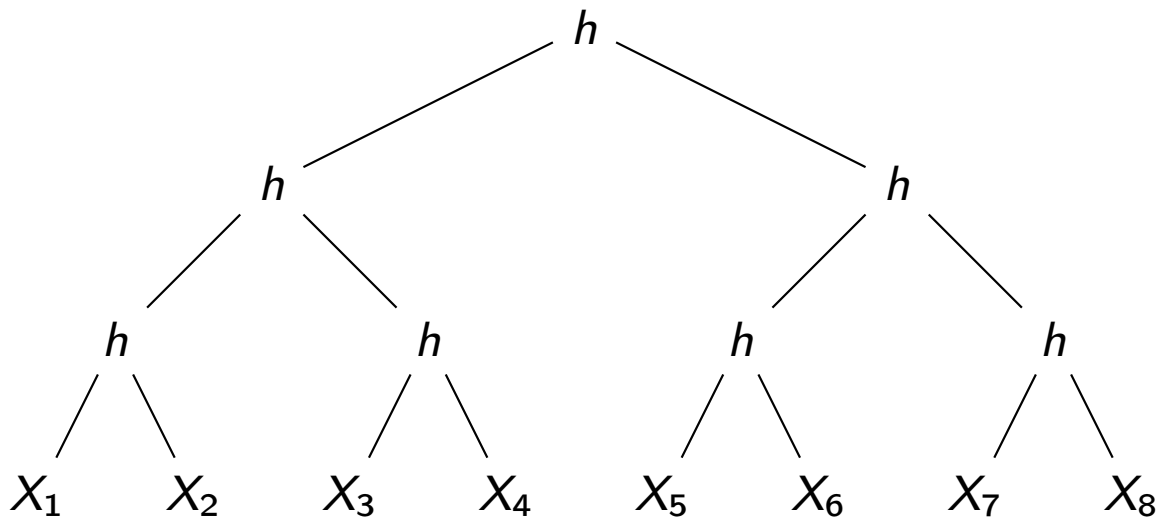
- Simple alternating algorithm for the optimization in a given tree-based format $\mathcal{T}_r^T$ which exploits the multilinearity of the parametrization.
  At each step, optimization over one parameter (learning problem with a linear model).
- Efficient strategy for rank adaptation based on higher order singular value decomposition.
- Tree adaptation using a stochastic algorithm, able to explore the set of possible trees and recover hidden structures of functions.

Example: supervised learning of a composition of functions

Consider a tree-structured composition of functions

$$u(X) = h(h(h(X_1, X_2), h(X_3, X_4)), h(h(X_5, X_6), h(X_7, X_8))),$$

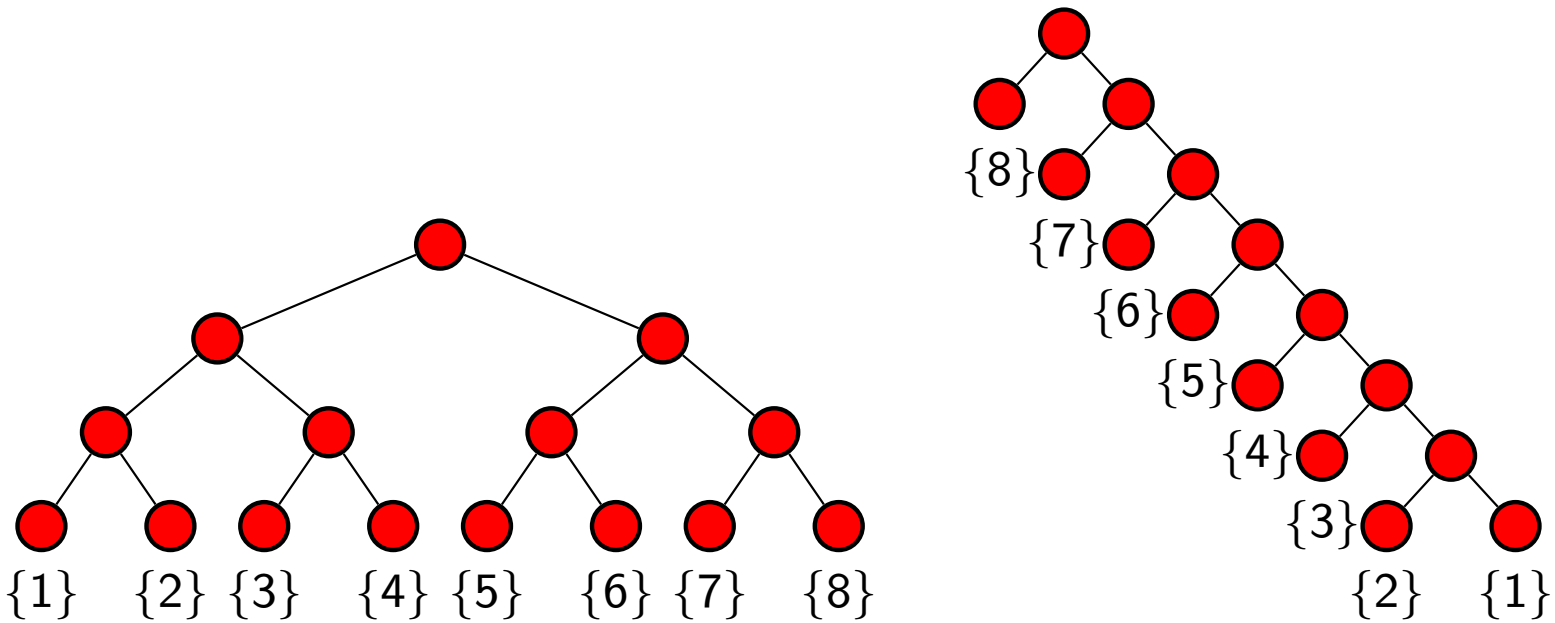
where $h(t, s) = 9^{-1}(2 + ts)^2$ is a bivariate function and where the $d = 8$ random variables $X_1, \ldots, X_8$ are independent and uniform on $[-1, 1]$.



We use approximation spaces such that u could (in principle) be recovered exactly for any choice of tree with a sufficiently high rank.

Example: supervised learning of a composition of functions

We consider two trees T^1 (coinciding with the structure of u) and T^2.



(a) Tree T^1 (b) Tree T^2

We start the learning algorithm from the tree and the associated families of trees $T^2_\sigma = \{\sigma(\alpha) : \alpha \in T^2\}$ obtained by applying a random permutation σ to T^2.

Behavior of the algorithm with $n = 10^5$

Iteration	rank r	$\varepsilon_{test}(v)$	$C(T, r)$
1	$(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)$	$3.38\,10^{-2}$	79
2	$(1,1,2,1,1,1,1,1,1,1,1,1,2,1,1)$	$2.95\,10^{-2}$	100
3	$(1,1,2,1,1,1,1,1,1,1,1,1,2,1,1)$	$2.95\,10^{-2}$	100
4	$(1,1,2,1,2,1,1,1,2,1,1,1,2,1,1)$	$2.45\,10^{-2}$	121
5	$(1,1,2,1,2,1,1,1,2,1,1,1,2,1,1)$	$2.45\,10^{-2}$	121
6	$(1,1,2,1,2,1,1,1,2,1,2,1,2,2,1)$	$1.85\,10^{-2}$	142
7	$(1,1,2,1,2,1,1,1,2,1,2,1,2,2,1)$	$1.85\,10^{-2}$	142
8	$(1,1,2,1,2,1,2,1,2,1,2,1,2,2,2)$	$8.97\,10^{-3}$	163
9	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$9.54\,10^{-3}$	188
10	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$8.89\,10^{-3}$	188
11	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$9.47\,10^{-3}$	188
12	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$8.87\,10^{-3}$	188
13	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$5.22\,10^{-3}$	188
14	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$3.97\,10^{-3}$	188
15	$(1,2,3,2,3,2,3,2,3,2,3,2,3,3,3)$	$1.55\,10^{-4}$	308
16	$(1,3,3,3,3,2,3,3,3,2,3,3,3,3,3)$	$1.18\,10^{-4}$	364
17	$(1,3,3,3,3,2,3,3,3,2,3,3,3,3,3)$	$1.18\,10^{-4}$	364

Example: learning a graphical model

Consider a truncated normal distribution with density

$$f(x)\mathrm{d}\mu(x) \sim \exp\left(-\frac{1}{2}x^T\Sigma^{-1}x\right)\mathbf{1}_{x\in\mathcal{X}},$$

and a matrix Σ such that $(X_1, X_3, X_4, X_6) \perp\!\!\!\perp (X_2, X_5)$ and , $X_4 \perp\!\!\!\perp (X_3, X_6)$.

Then f is represented by the following graphical model:

$$f(x) = f_{4,1}(x_4, x_1)f_{1,3,6}(x_1, x_3, x_6)f_{2,5}(x_2, x_5)$$

.

Example: learning a graphical model

n	Risk $\times 10^{-2}$	L^2-error	T	$C(T, r)$
10^2	$[-5.50, 119]$	$[0.53, 4.06]$	Fig. (a)	$[311, 311]$
10^3	$[-7.29, -5.93]$	$[0.22, 0.47]$	Fig. (b)	$[311, 637]$
10^4	$[-7.60, -6.85]$	$[0.11, 0.33]$	Fig. (c)	$[521, 911]$
10^5	$[-7.68, -7.66]$	$[0.04, 0.07]$	Fig. (c)	$[911, 1213]$
10^6	$[-7.70, -7.69]$	$[0.01, 0.01]$	Fig. (c)	$[1283, 1546]$

Table: Ranges over 10 trials

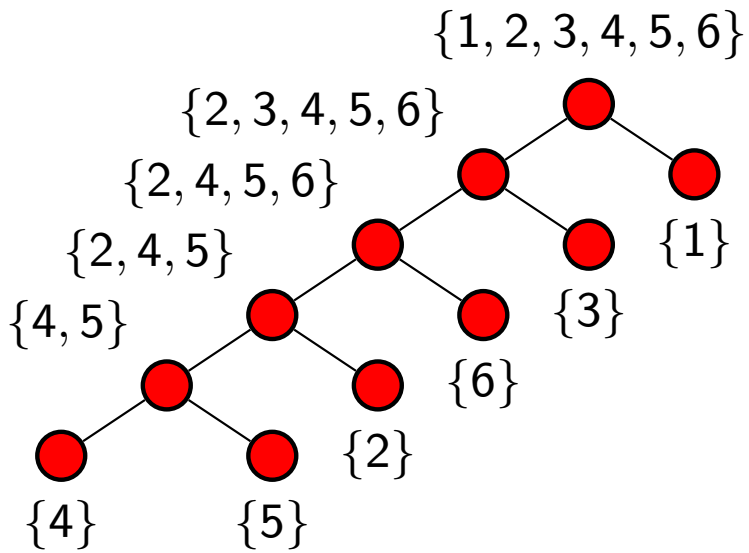
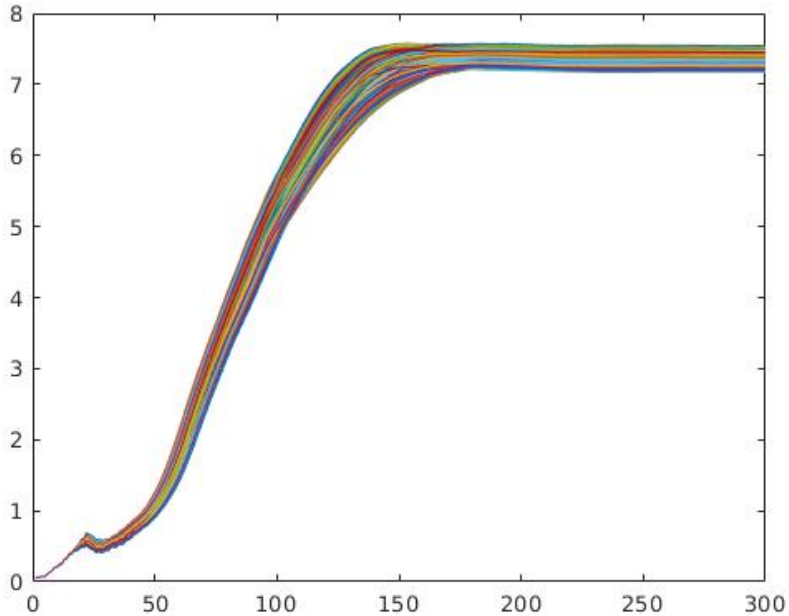


Figure: (a) Best tree over 10 trials for $n = 10^2$

Example: Pressure in a rocket booster

Modelling of the stochastic process $p(t, \omega)$.



Truncated Karhunen-Loeve expansion

$$p(t, \omega) \approx m(t) + \sum_{i=1}^{d} \sigma_i X_i(\omega) \varphi_i(t)$$

Objective: estimate the density $f(x)$ of $X = (X_1, \ldots, X_d)$

Example: Pressure in a rocket booster

L^2-loss density estimation using polynomial or polynomial multiwavelets approximation spaces.

Approximation spaces	$\mathcal{R}_{\text{test}}(v)$	$C(T, r)$	$\max_\alpha r_\alpha$
Polynomials	$-5.11\,10^3$	8251	26
Polynomial multiwavelets	$-9.74\,10^3$	9214	21

Table: Tree-based tensor format

Approximation spaces	$\mathcal{R}_{\text{test}}(v)$	$C(T, r)$
Polynomials	$-1.35\,10^3$	131
Polynomial multiwavelets	$-1.64\,10^3$	197

Table: Rank-one approximation, independence hypothesis.

Example: Pressure in a rocket booster

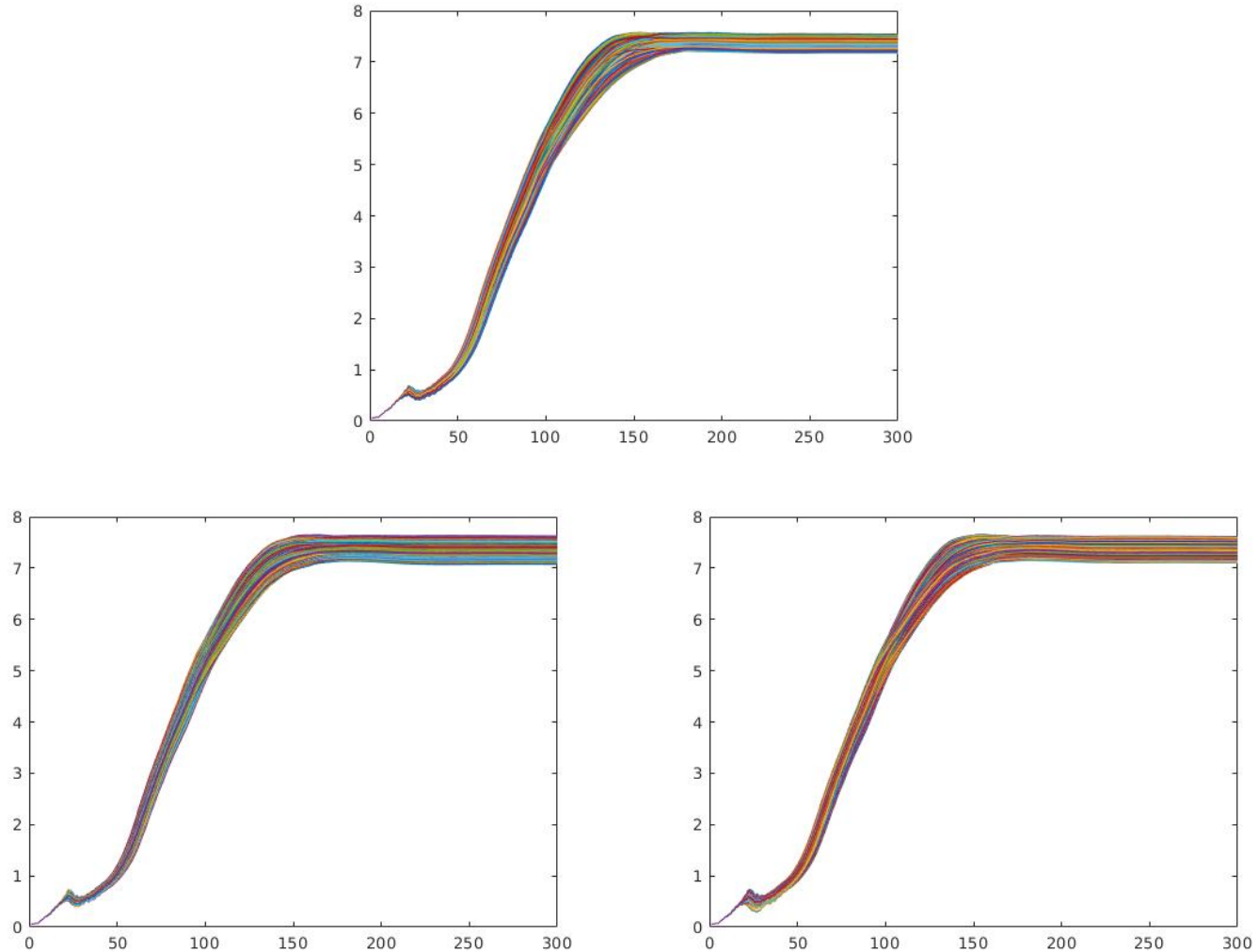


Figure: Samples from the true distribution (top), and the one estimated with independence hypothesis (bottom left) or using tree-based format (bottom right)

References I

A. Nouy.

Low-rank methods for high-dimensional approximation and model order reduction.
In P. Benner, A. Cohen, M. Ohlberger, and K. Willcox (eds.), Model Reduction and Approximation: Theory and Algorithms. SIAM, Philadelphia, PA, 2016.

M. Chevreuil, R. Lebrun, A. Nouy, and P. Rai.

A least-squares method for sparse low rank approximation of multivariate functions.
SIAM/ASA Journal on Uncertainty Quantification, 3(1):897–921, 2015.

E. Grelier, A. Nouy, M. Chevreuil.

Learning with tree-based tensor formats.
Arxiv eprints, Nov. 2018.

A. Nouy.

Higher-order principal component analysis for the approximation of tensors in tree-based low-rank formats.
Numerische Mathematik, 141(3):743–789, Mar 2019.