Model Order Reduction Summer School 2019,
Eindhoven, Sep. 23-27, 2019

Approximation with tree tensor networks

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

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

High-dimensional problems in mechanics and physics

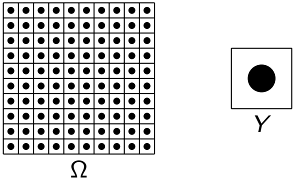
- Navier Stokes equation

$$u(x, t)$$

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

- Multiscale problems

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



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.

Low-dimensional problems as high-dimensional problems

Consider a function $u(x)$ defined on $[0,1)$.

- By subdividing $[0,1)$ into N intervals of equal length, u can be identified with a bivariate function
$$u(x) = v(i, y),$$
where
$$x = N^{-1}(i+y), \quad i \in \{0, \dots, N-1\}, \quad y \in [0,1)$$
- If $N = 2^d$ with $d \in \mathbb{N}$, then $i \in \{0, \dots, 2^d - 1\}$ can be written in base 2
$$i = \sum_{k=0}^{d-1} i_k 2^k$$
and u can be identified with a $(d+1)$-dimensional function
$$v(i_0, \dots, i_{d-1}, y), \quad i_\nu \in \{0,1\}, \quad y \in [0,1).$$

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.
- 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 how fast $e_n(u)$ converges for a certain class of functions, e.g.

  $$e_n(u) \leq M\gamma(n)^{-1}$$

  where $\gamma(n)$ is a strictly increasing function, or determine the complexity $n = n(\epsilon, u) \geq \gamma^{-1}(\epsilon/M)$ for having $e_n(u) \leq \epsilon$.
- Characterize approximation classes, i.e. sets of functions for which the approximation tool has a certain performance, e.g.

  $$\mathcal{A}^\gamma = \{u : \sup_n \gamma(n) e_n(u) < +\infty\}$$

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

What can we expect from an ideal approximation tool ?

For a set of functions K in a normed vector space X, the Kolmogorov n-width of K is

$$d_n(K)_X = \inf_{\dim(X_n)=n} \sup_{u \in K} \inf_{v \in X_n} \|u - v\|_X$$

where the infimum is taken over all linear subspaces of dimension n.

$d_n(K)_X$ measures how well the set of functions K can be approximated by a n-dimensional space.

It measures the ideal performance that we can expect from linear approximation methods.

The curse of dimensionality

- For $X = L^2(\mathcal{X})$ with $\mathcal{X} = (0,1)^d$ or $\mathcal{X} = \mathbb{T}^d$, and K the unit ball of the Sobolev space $H^k(\mathcal{X})$,
$$d_n(K)_X \sim n^{-k/d}$$
this optimal rate being achieved with splines or trigonometric polynomials.

  We observe the curse of dimensionality: deterioration of the rate of convergence when d increases, exponential growth with d of the required complexity for reaching a given accuracy.
- For $X = L^2(\mathcal{X})$ with $\mathcal{X} = \mathbb{T}^d$, and K the unit ball of the mixed Sobolev space $H^k_{mix}(\mathcal{X})$,
$$d_n(K)_X \sim n^{-k} \log(n)^{k(d-1)},$$
this rate being achieved by sparse tensors (hyperbolic cross approximation).

  The curse of dimensionality is still present.
- For $X = L^\infty(\mathcal{X})$ with $\mathcal{X} = (0,1)^d$ and $K = \{v \in C^\infty(\mathcal{X}) : \sup_\alpha \|D^\alpha v\|_{L^\infty} < \infty\}$,
$$\min\{n : d_n(K)_X \le 1/2\} \ge c2^{d/2}$$
No blessing of smoothness !

How to beat the curse of dimensionality ?

- Similar results hold for nonlinear widths that measure the ideal performance of nonlinear approximation tools for standard regularity classes.
- No (reasonable) approximation tool is able to overcome the curse of dimensionality for these standard regularity classes.
- The key is to consider classes of functions with specific low-dimensional structures and to propose approximation formats (models) which exploit these structures (application-dependent).

Outline

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.

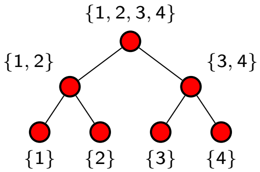
$$f(g_{1,2}(x_1, x_2), g_{3,4}(x_3, x_4), \ldots)$$

More compositions... deep neural networks

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

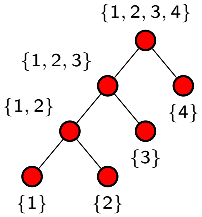
- Convolutional networks, sparse transformations with sparsity induced by a balanced tree

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



- 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.

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 $\alpha \subset \{1, \dots, d\} := D$, $v(x)$ 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
$$v(x) = \sum_{k=1}^{r_\alpha} v_k^\alpha(x_\alpha) w_k^{\alpha^c}(x_{\alpha^c})$$

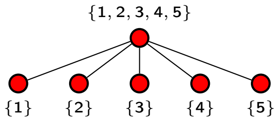
Tree based tensor formats

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

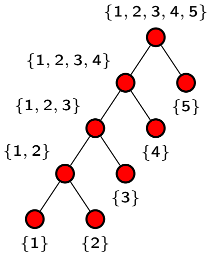
$$\mathcal{T}_r^T(\mathcal{H}) = \{v \in \mathcal{H} : \mathrm{rank}_\alpha(v) \leq r_\alpha, \alpha \in T\}.$$

  with $\mathcal{H}$ a space of multivariate functions.

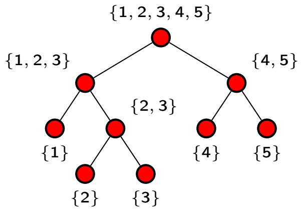
- In the particular case where T is a dimension partition tree, $\mathcal{T}_r^T$ is a tree-based tensor format.



Tucker



Tensor Train



Hierarchical Tucker

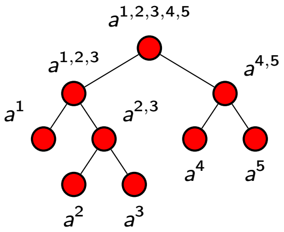
Tree-based formats as tensor networks

Consider a tensor space $\mathcal{H} = \mathcal{H}^1 \otimes \ldots \otimes \mathcal{H}^d$ of functions in $L^2_\mu(\mathcal{X})$, and let $\{\phi^\nu_{i_\nu} : i_\nu \in I^\nu\}$ be a basis of $\mathcal{H}^\nu \subset L^2_{\mu_\nu}(\mathcal{X}_\nu)$, typically polynomials, wavelets...

A function v in $\mathcal{T}_r^T(\mathcal{H}) = \{v \in \mathcal{H} : \mathrm{rank}_T(v) \le r\}$ admits an explicit representation

$$v(x) = \sum_{\substack{i_\alpha \in I^\alpha \\ \alpha \in \mathcal{L}(T)}} \sum_{\substack{1 \le k_\beta \le r_\beta \\ \beta \in T}} \prod_{\alpha \in T \setminus \mathcal{L}(T)} a^\alpha_{(k_\beta)_{\beta \in S(\alpha)}, k_\alpha} \prod_{\alpha \in \mathcal{L}(T)} a^\alpha_{i_\alpha, k_\alpha} \phi^\alpha_{i_\alpha}(x_\alpha)$$

where each parameter a^α is in a tensor space $\mathbb{R}^{K^\alpha}$.



Number of parameters (representation complexity)

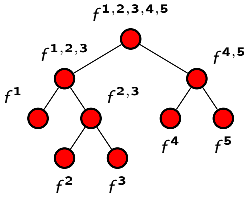
$$C(T, r, \mathcal{H}) = \sum_{\alpha \in T \setminus \mathcal{L}(T)} r_\alpha \prod_{\beta \in S(\alpha)} r_\beta + \sum_{\alpha \in \mathcal{L}(T)} r_\alpha \dim(\mathcal{H}^\alpha).$$

Tree-based tensor format as a deep neural network

By identifying a tensor $a^{(\alpha)} \in \mathbb{R}^{n_1 \times \ldots \times n_s \times r_\alpha}$ with a $\mathbb{R}^{r_\alpha}$-valued multilinear function

$$f^{(\alpha)} : \mathbb{R}^{n_1} \times \ldots \times \mathbb{R}^{n_s} \to \mathbb{R}^{r_\alpha},$$

a function v in $\mathcal{T}_r^T$ admits a representation as a tree-structured composition of multilinear functions $\{f^{(\alpha)}\}_{\alpha \in T}$.



$$v(x) = f^D(f^{1,2,3}(f^1(\Phi^1(x_1)), f^{2,3}(f^2(\Phi^2(x_2)), f^3(\Phi^3(x_3))), f^{4,5}(f^4(\Phi^4(x_4)), f^5(\Phi^5(x_5))))$$

where $\Phi^\nu(x_\nu) = (\phi^\nu_{i_\nu}(x_\nu))_{i_\nu \in I^\nu} \in \mathbb{R}^{\#I^\nu}$.

It corresponds to a deep network with a sparse architecture (given by T), a depth bounded by $d-1$, and width at level ℓ related to the α-ranks of the nodes α of level ℓ.

Outline

Singular value decompositions of multivariate functions

We consider a multivariate function u in $L^2_\mu(\mathcal{X})$, where $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_d$ is equipped with a product measure $\mu = \mu_1 \otimes \ldots \otimes \mu_d$.

Consider a subset of variables α and its complementary subset $\alpha^c = D \setminus \alpha$.

$u(x_1, \ldots, x_d)$ can be identified with a bivariate function $u(x_\alpha, x_{\alpha^c})$ in $L^2_{\mu_\alpha \otimes \mu_{\alpha^c}}(\mathcal{X}_\alpha \times \mathcal{X}_{\alpha^c})$ which admits a singular value decomposition

$$u(x_\alpha, x_{\alpha^c}) = \sum_{k=1}^{\mathrm{rank}_\alpha(u)} \sigma_k^\alpha v_k^\alpha(x_\alpha) v_k^{\alpha^c}(x_{\alpha^c})$$

The problem of best approximation of u by a function with α-rank r_α,

$$\min_{\mathrm{rank}_\alpha(v) \leq r_\alpha} \|u - v\|^2 := e^\alpha_{r_\alpha}(u)^2,$$

admits as a solution the truncated singular value decomposition u_{r_α} of u

$$u_{r_\alpha}(x_\alpha, x_{\alpha^c}) = \sum_{k=1}^{r_\alpha} \sigma_k^\alpha v_k^\alpha(x_\alpha) v_k^{\alpha^c}(x_{\alpha^c})$$

where $\{v_1^\alpha, \ldots, v_{r_\alpha}^\alpha\}$ are the r_α α-principal components of u.

α-principal subspaces and associated projections

The subspace of principal components

$$U_\alpha = span\{v_1^\alpha, \dots, v_{r_\alpha}^\alpha\}$$

is such that

$$u_{r_\alpha}(\cdot, x_{\alpha^c}) = \mathcal{P}_{U_\alpha} u(\cdot, x_{\alpha^c})$$

where $\mathcal{P}_{U_\alpha}$ is the orthogonal projection onto U_α.

It is solution of

$$\min_{\dim(U_\alpha)=r_\alpha} \|u - \mathcal{P}_{U_\alpha} u\|^2$$

that is

$$\min_{\dim(U_\alpha)=r_\alpha} \int \|u(\cdot, x_{\alpha^c}) - \mathcal{P}_{U_\alpha} u(\cdot, x_{\alpha^c})\|^2_{L^2_{\mu_\alpha}} \, d\mu_{\alpha^c}(x_{\alpha^c}).$$

Linear widths for multivariate functions

Consider the set of functions

$$K_\alpha(u) = \{u(\cdot, x_{\alpha^c}) : x_{\alpha^c} \in \mathcal{X}_{\alpha^c}\} \subset L^2_{\mu_\alpha}(\mathcal{X}_\alpha)$$

and let ν_{α^c} be the push-forward measure of μ_{α^c} over $K_\alpha(u)$ through the map $x_{\alpha^c} \mapsto u(\cdot, x_{\alpha^c})$.

The best approximation error $e^\alpha_{r_\alpha}(u)$ is such that

$$e^\alpha_{r_\alpha}(u)^2 = \min_{\dim(U_\alpha)=r_\alpha} \int_{K_\alpha(u)} \|v - \mathcal{P}_{U_\alpha} v\|^2_{L^2_{\mu_\alpha}} d\nu_{\alpha^c}(v)$$

and defines a linear width of the set $K_\alpha(u)$ which measures how well it can be approximated by a r_α dimensional space U_α. It quantifies the ideal performance of a linear approximation method in $L^2_{\mu_\alpha}(\mathcal{X}_\alpha)$ in a mean-square sense.

Linear widths for multivariate functions

Assuming μ is finite,

$$e^{\alpha}_{r_\alpha}(u) \lesssim \min_{\dim(U_\alpha)=r_\alpha} \sup_{v\in K_\alpha(u)} \|v - \mathcal{P}_{U_\alpha} v\|_{L^2_{\mu_\alpha}} = d_{r_\alpha}(K_\alpha(u))_{L^2_{\mu_\alpha}},$$

this upper bound being the Kolmogorov r_α-width of $K_\alpha(u)$ in $L^2_{\mu_\alpha}(\mathcal{X}_\alpha)$.

Furthermore, since

$$e^{\alpha}_{r_\alpha}(u) = e^{\alpha^c}_{r_\alpha}(u),$$

we have

$$e^{\alpha}_{r_\alpha}(u) \leq \min\left\{ d_{r_\alpha}(K_\alpha(u))_{L^2_{\mu_\alpha}}, d_{r_\alpha}(K_{\alpha^c}(u))_{L^2_{\mu_{\alpha^c}}} \right\}$$

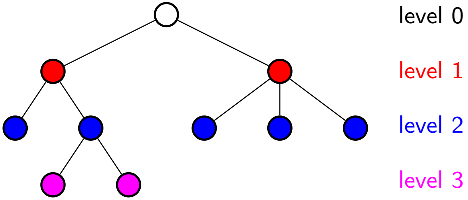
Singular value decomposition in tree-based formats

The notion of singular value decomposition can be extended (in different ways) to higher-order tensors.

Given a dimension tree T, for each $\alpha \in T$, we let U_α be a r_α-dimensional subspace of $L^2_{\mu_\alpha}$ and define

$$u_r = \mathcal{P}^{(L)}\mathcal{P}^{(L-1)} \dots \mathcal{P}^{(1)} u \quad \text{with} \quad \mathcal{P}^{(\ell)} = \prod_{\substack{\alpha \in T \\ \text{level}(\alpha)=\ell}} \mathcal{P}_{U_\alpha}$$

where we apply to u a sequence of projections ordered by increasing level in the tree (from the root to the leaves). Here $L = \max_{\alpha \in T} \text{level}(\alpha)$.



Singular value decomposition in tree-based formats

We can prove that $u_r \in \mathcal{T}_r^T$ with $r = (r_\alpha)_{\alpha \in T}$, so that

$$\|u - u_r\| \geq \min_{v \in \mathcal{T}_r^T} \|u - v\| := e_r^T(u)$$

and

$$\|u - u_r\|^2 \leq \sum_{\alpha \in T \setminus \{D\}} \|u - \mathcal{P}_{U_\alpha} u\|^2.$$

Taking for U_α the α-principal subspaces, we then have

$$e_r^T(u)^2 \leq \|u - u_r\|^2 \leq \sum_{\alpha \in T \setminus \{D\}} e_{r_\alpha}^\alpha(u)^2$$

Noting that for all α,

$$e_{r_\alpha}^\alpha(u) = \min_{\text{rank}_\alpha(v) \leq r_\alpha} \|u - v\| \leq \min_{v \in \mathcal{T}_r^T} \|u - v\|,$$

we obtain an instance optimality result

$$e_r^T(u) \leq \|u - u_r\| \leq \sqrt{\#T} e_r^T(u)$$

with $d + 1 \leq \#T \leq 2d - 1$. For a binary tree, $\#T = 2d - 1$.

Singular value decomposition in tree-based formats

For a desired precision ϵ, if the α-ranks r_α are chosen such that

$$\|u - \mathcal{P}_{U_\alpha} u\| \leq \frac{\epsilon}{\sqrt{\# T}} \|u\|,$$

we obtain an approximation u_r such that

$$\|u - u_r\| \leq \epsilon \|u\|.$$

This provides an algorithm based on classical singular value decompositions for "compressing" a tensor at a given precision.

Outline

Approximation power of tree tensor networks

We want to quantify the approximation error

$$\min_{v \in \mathcal{T}_r^T} \|u - v\|$$

for a function u in a given function class, i.e. study the expressive power of tree tensor networks, and compare it with other approximation tools.

Expressive power of tree tensor networks

- For standard regularity classes, they perform almost as well as standard approximation tools.

  For example, for $u \in H^k_{mix}((0,1)^d)$, $K_\alpha(u) \subset H^k_{mix}((0,1)^{\#\alpha})$ for any α. From bounds of Kolmogorov widths of Sobolev balls

  $$e^{\alpha}_{r_\alpha}(u) \le d_{r_\alpha}(K_\alpha(u)) \lesssim r_\alpha^{-k} \log(r_\alpha)^{k(\#\alpha-1)}$$

  we obtain that the complexity to achieve a precision ϵ (with binary trees) is

  $$n(\epsilon, u) \lesssim \epsilon^{-3/k} \log(\epsilon^{-1})^d d^{1+3/(2k)} \quad \text{up to powers of } \log(\epsilon^{-1})$$

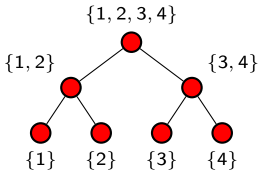
  Performs almost as well as hyperbolic cross approximation (sparse tensors).

- Similar results in [Schneider and Uschmajew 2014] using results on bilinear approximation [Temlyakov 1989]. See also [Griebel and Harbrecht 2019]

Expressive power of tree tensor networks
[with M. Bachmayr and R. Schneider]

- But they can perform much better for non standard classes of functions, e.g. a tree-structured composition of regular functions $\{f_\alpha : \alpha \in T\}$, see [Mhaskar, Liao, Poggio 2016] for deep neural networks.

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



Assuming that the functions $f_\alpha \in W^{k,\infty}$ with $\|f_\alpha\|_{L^\infty} \le 1$ and $\|f_\alpha\|_{W^{k,\infty}} \le B$, the complexity to achieve an accuracy ϵ

$$n(\epsilon, u) \lesssim \epsilon^{-3/k}(L+1)^3 B^{3L} d^{1+3/2k}$$

with $L = \log_2(d)$ for a balanced tree and $L + 1 = d$ for a linear tree.

- Bad influence of the depth through the norm B of functions f_α (roughness).
- For $B \le 1$ (and even for 1-Lipschitz functions), the complexity only scales polynomially in d: no curse of dimensionality !

Expressive power of tree tensor networks

- A function in canonical format (shallow 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 d.

  Deep is better !

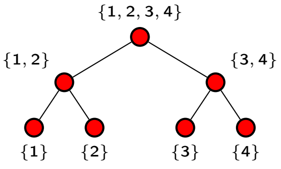
  For a balanced or linear binary tree T, the subset of tensors v in $\mathcal{T}_r^T(\mathbb{R}^{n\times\dots\times n})$ with canonical rank less than $\min\{n,r\}^{d/2}$ is of Lebesgue measure 0 [Cohen et al. 2016, Khrulkov et al 2018]

Influence of the tree

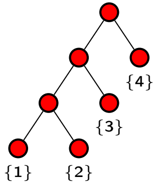
- For some functions, the performance does not depend so much on the tree. For example, an additive function

$$f_1(x_1) + \ldots + f_d(x_d)$$

  has α-ranks equal to 2 whatever $\alpha \subset D$.
- But usually, different trees lead to different performances.



T^B (Balanced tree)



T^L (Linear tree)

  - If $\text{rank}_{T^L}(u) \leq r$ then $\text{rank}_{T^B}(u) \leq r^2$
  - If $\text{rank}_{T^B}(u) \leq r$ then $\text{rank}_{T^L}(u) \leq r^{\log_2(d)/2}$
- But a typical function in $\mathcal{T}_r^T$ may admit a representation complexity exponential in d when using another tree.

Influence of the tree

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

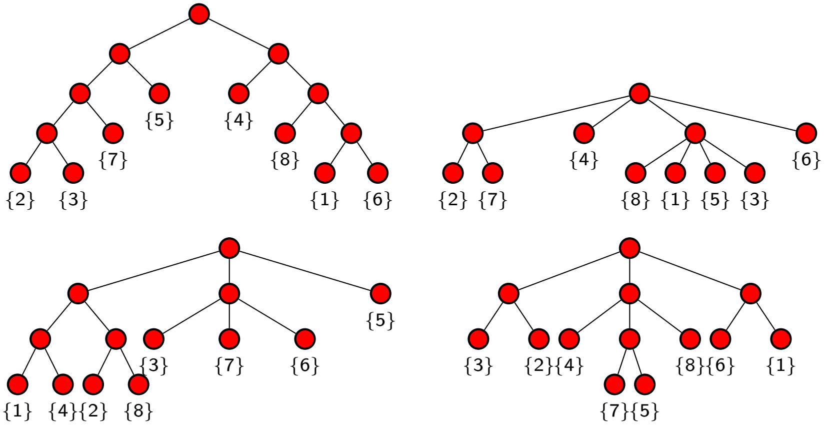
$$f(x) = f_1(x_1) f_{2|1}(x_2|x_1) \dots f_{d|d-1}(x_d|x_{d-1})$$

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

- With the linear tree T containing interior nodes $\{1,2\}, \{1,2,3\}, \dots, \{1,\dots,d-1\}$, 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 $T_\sigma = \{\sigma(\alpha) : \alpha \in T\}$ obtained by applying permutation $\sigma = (1,3,\dots,d-1,2,4,\dots,d)$ to the tree T, the storage complexity in tree-based format is also exponential in d.

Approximation properties of tree tensor networks

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



Can be solved with stochastic algorithms (requires some heuristics)
[Grelier, Nouy and Chevreuil 2018].

Approximation of functions through tensorization

For a function $u(x)$ defined for $x \in [0, 1)$, we introduce the corresponding multivariate function v defined on $\{0, ..., b-1\}^d \times [0, 1)$ such that

$$u(x) = v(i_0, \ldots, i_{d-1}, y)$$

where

$$x = b^{-d} y + b^{-d} \sum_{k=0}^{d-1} i_k b^k.$$

- This allows the identification (through a linear isometry)

$$L^2(0,1) = \mathbb{R}^b \ldots\ldots \mathbb{R}^b \otimes L^2(0,1).$$

- In practice, introduction of an approximation space $S_p \subset L^2(0,1)$ (e.g. polynomial space) and approximations in

$$V_{b,d,p} = \mathbb{R}^b \otimes \ldots \otimes \mathbb{R}^b \otimes S_p$$

  and use of tree-based formats in $V_{b,d,p}$.
- For example, $V_{2,d,0}$ corresponds to the space of piecewise constant functions on a uniform mesh with 2^d elements.

Approximation of functions through tensorization

Exploiting low-rank structures of the tensorized function allows to achieve better performance than splines on adapted meshes for functions with singularities or multiscale functions [Kazeev and Schwab 2015 , Kazeev et al. 2017].

- For $u(x) = x^{\alpha}$, $0 < \alpha \leq 1$,
  - a piecewise constant approximation on a uniform mesh with n elements gives a convergence in $O(n^{-\alpha})$ in L^{∞},
  - a piecewise constant approximation on an optimal mesh with n elements gives a convergence in $O(n^{-1})$ in L^{∞},
  - a piecewise constant approximation on a uniform mesh with 2^d elements exploiting low-rank structures gives an exponential convergence in $O(\beta^{-n})$, where n is the complexity of the representation.
- For $u(x) = e^{zx}$, $z \in \mathbb{C}$,

$$v(i_0, \dots, i_{d-1}, y) = u_1(i_0), \dots u_d(i_{d-1}) u_{d+1}(y), \quad \text{with } u_k(t) = e^{ztb^{k-d}},$$

is a rank-one function whatever z.

Approximation of functions through tensorization

A promising route for high-resolution simulations in low-dimension.

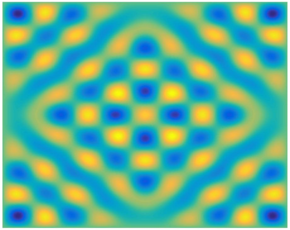


Figure: Scattering problem: tensorization with base $b = 2$, piecewise constant approximation, storage complexity at precision 10^{-3} (resp. 10^{-5}) goes from 260100 to 3532 (resp. 6170) by exploiting low-rank structures.

Outline

Statistical learning

Two typical tasks of statistical learning are to

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

Risk

A classical approach is to introduce a risk functional $\mathcal{R}(v)$ whose minimizer over the set of functions v is the target function u and such that

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

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

The risk is defined as an expectation

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

where γ is called a contrast (or loss) function.

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

Risk

- Variational methods for PDEs [Eigel et al 2018]: with Z uniformly distributed on $D = (0,1)^d$ and a risk

$$\mathcal{R}(v) = \mathbb{E}(|\nabla v(Z)|^2) - 2v(Z)f(Z))$$

the target function u in H_0^1 is such that

$$-\Delta u = f \quad \text{on} \quad D,$$

and

$$\mathcal{R}(v) - \mathcal{R}(u) = \|v - u\|_{H_0^1}^2$$

Outline

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 $\nearrow$.
- Methods should be proposed for the selection of a model class taking the best from the available information.

Learning algorithm for tree tensor networks

A function v in the model class $\mathcal{T}_r^T(\mathcal{H})$ has a representation $v(x) = \Psi(x)((a^\alpha)_{\alpha\in T})$ where each parameter a^α is in a tensor space $\mathbb{R}^{K^\alpha}$ and $\Psi(x)$ is a multilinear map.

The empirical risk minimization problem over the nonlinear model class $\mathcal{T}_r^T$

$$\min_{(a^\alpha)_{\alpha\in T}} \frac{1}{n}\sum_{i=1}^{n} \gamma(\Psi(\cdot)((a^\alpha)_{\alpha\in T}), z_i)$$

can be solved using an alternating minimization algorithm, solving at each step an empirical risk minimization problem with a linear model

$$\Psi(x)((a^\alpha)_{\alpha\in T}) = \sum_{k\in K^\alpha} \Psi_k^\alpha(x) a_k^\alpha$$

with functions $\Psi_k^\alpha(x)$ depending on fixed parameters a^β, $\beta \neq \alpha$.

- In a L^2 setting, possible re-parametrization for having orthonormal functions $\Psi_k^\alpha(x)$.
- Sparsity in the tensors a^α can be exploited in different ways, e.g. by proposing different sparsity patterns and use a model selection technique (e.g. based on validation).
- For a leaf node ν, the approximation space $\mathcal{H}^\nu$ can be selected from a candidate sequence of spaces $\mathcal{H}_0^\nu \subset \ldots \subset \mathcal{H}_L^\nu \subset \ldots$

Learning algorithm for tree tensor networks

Selection an optimal model class $\mathcal{T}_r^T(\mathcal{H})$ is a combinatorial problem.

An algorithm is proposed in [Grelier, Nouy, Chevreuil 2018] that performs adaptations of the tree T (architecture), the rank r (widths) and the approximation space $\mathcal{H}$.

Start with an initial tree T and learn an approximation $v \in \mathcal{T}_r^T(\mathcal{H})$ with rank $r = (1, ..., 1)$. Then repeat

- Increase some ranks r_α based on estimates of truncation errors

$$\min_{\text{rank}_\alpha(v) \leq r_\alpha} \mathcal{R}(v) - \mathcal{R}(u)$$

- Learn an approximation v in $\mathcal{T}_r^T(\mathcal{H})$, with adaptive selection of $\mathcal{H}$
- Optimize the tree for reducing the storage complexity of v (stochastic algorithm using a suitable distribution over the set of trees)

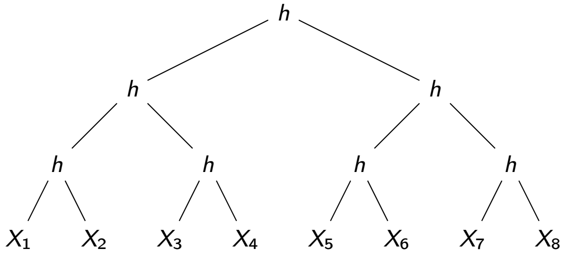
$$\min_T C(T, \text{rank}_T(v), \mathcal{H})$$

Example in supervised learning: 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 $X_1, \dots, X_8$ are independent and uniform on $[-1, 1]$.

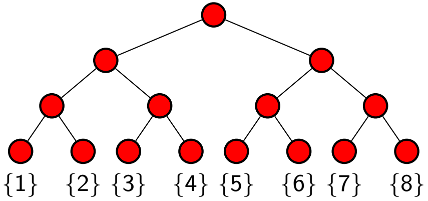


We use polynomial approximation spaces $\mathcal{H}$ (with adaptive selection of the degree), so that function u could (in principle) be recovered exactly for any choice of tree with a sufficiently high rank.

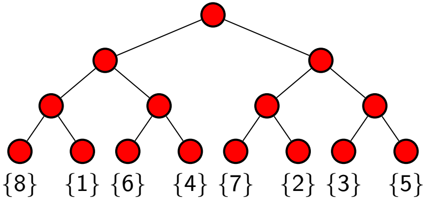
Example in supervised learning: composition of functions

We consider the tree T^1 coinciding with the structure of u, for which

$$C(T^1, \mathrm{rank}_{T^1}(u), \mathcal{H}) = 2427$$



(a) Tree T^1



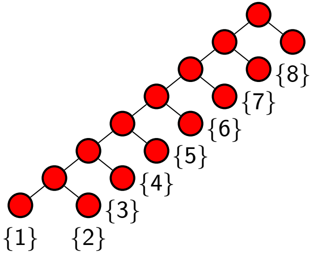
(b) Tree T^1_σ

By considering a permutation $T^1_\sigma = \{\sigma(\alpha) : \alpha \in T^1\}$ of T^1, with $\sigma = (8, 1, 6, 4, 7, 2, 3, 5)$, we have a complexity

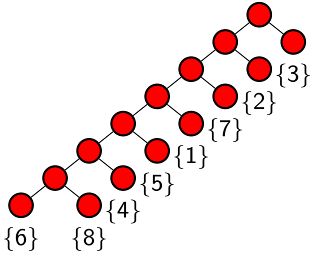
$$C(T^1_\sigma, \mathrm{rank}_{T^1_\sigma}(u), \mathcal{H}) \geq 9.10^6$$

Example in supervised learning: composition of functions

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



(c) Tree T^2



(d) Tree T^2_σ for $\sigma = (6, 8, 4, 5, 1, 7, 2, 3)$

Example in supervised learning: composition of functions

Behavior of the algorithm with a sample size $n = 10^5$

Iteration	rank r	$\varepsilon_{test}(v)$	$C(T, r, \mathcal{H})$
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	$(1,1,2,1,2,1,1,1,2,1,1,1,2,1,1)$	$2.45\,10^{-2}$	121
4	$(1,1,2,1,2,1,1,1,2,1,2,1,2,2,1)$	$1.85\,10^{-2}$	142
5	$(1,1,2,1,2,1,2,1,2,1,2,1,2,2,2)$	$8.97\,10^{-3}$	163
6	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$8.89\,10^{-3}$	188
7	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$8.87\,10^{-3}$	188
8	$(1,2,2,2,2,2,2,2,2,1,2,2,2,2,2)$	$3.97\,10^{-3}$	188
9	$(1,2,3,2,3,2,3,2,3,2,3,2,3,3,3)$	$1.55\,10^{-4}$	308
10	$(1,3,3,3,3,2,3,3,3,2,3,3,3,3,3)$	$1.18\,10^{-4}$	364
11	$(1,3,4,3,4,2,4,3,4,2,4,3,4,4,4)$	$6.65\,10^{-6}$	520
12	$(1,3,5,3,5,3,5,3,5,3,5,3,5,5,5)$	$1.19\,10^{-6}$	723
13	$(1,4,5,4,5,3,5,4,5,3,5,4,5,5,5)$	$1.72\,10^{-7}$	865
14	$(1,4,6,4,6,3,6,4,6,3,6,4,6,6,6)$	$1.47\,10^{-8}$	1113
15	$(1,5,6,5,6,3,6,5,6,3,6,5,6,6,6)$	$7.02\,10^{-9}$	1311
16	$(1,5,7,5,7,3,7,5,7,3,7,5,7,7,7)$	$1.27\,10^{-10}$	1643

Example in supervised learning: composition of functions

Behavior of the algorithm for different sample sizes n.

n	$\hat{\mathbb{P}}(T = T^1)$	$\varepsilon_{test}(v)$	$C(T, r, \mathcal{H})$
10^3	90%	$[1.75\,10^{-5}, 1.75\,10^{-4}]$	$[360, 1062]$
10^4	90%	$[2.15\,10^{-8}, 4.10\,10^{-3}]$	$[185, 2741]$
10^5	100%	$[4.67\,10^{-15}, 8.92\,10^{-3}]$	$[163, 2594]$

Table: training sample size n, estimation of the probability of obtaining the ideal tree T^1 and ranges (over the 10 trials) for the test error, and the storage complexity.

Outline

Active learning based on principal component analysis

We consider the least-squares regression setting for supervised learning.

We consider the context of active learning for the approximation of $u(X)$, where samples of X can be chosen.

For the construction of an approximation in the tree-based format $\mathcal{T}_r^T$, we will determine for each node α in T approximations of α-principal subspaces of u, from structured samples of X.

Empirical principal component analysis

For $\alpha \subset D = \{1, \dots, d\}$, a subspace of α-principal components of $u(X)$ is solution of

$$\min_{\dim(U_\alpha)=r_\alpha} \mathbb{E}\left(\| u(\cdot, X_{\alpha^c}) - \mathcal{P}_{U_\alpha} u(\cdot, X_{\alpha^c}) \|^2_{L^2_{\mu_\alpha}(\mathcal{X}_\alpha)} \right)$$

where u is seen as a function-valued random variable

$$u(\cdot, X_{\alpha^c}) \in L^2_{\mu_\alpha}(\mathcal{X}_\alpha).$$

It can be estimated using i.i.d. samples $u(\cdot, x^j_{\alpha^c})$ of this random variable and by solving

$$\min_{\dim(U_\alpha)=r_\alpha} \frac{1}{N_\alpha} \sum_{j=1}^{N_\alpha} \| u(\cdot, x^j_{\alpha^c}) - \mathcal{P}_{U_\alpha} u(\cdot, x^j_{\alpha^c}) \|^2_{L^2_{\mu_\alpha}(\mathcal{X}_\alpha)}$$

where $\{x^j_{\alpha^c}\}_{j=1}^{N_\alpha}$ are i.i.d. samples of the group of variables X_{α^c}.

Empirical principal component analysis

In practice, we determine the principal subspaces of an approximation u_α of u by solving

$$\min_{\dim(U_\alpha)=r_\alpha} \frac{1}{N_\alpha} \sum_{j=1}^{N_\alpha} \| u_\alpha(\cdot, x^j_{\alpha^c}) - \mathcal{P}_{U_\alpha} u_\alpha(\cdot, x^j_{\alpha^c}) \|^2_{L^2_{\mu_\alpha}(\mathcal{X}_\alpha)}$$

For a given value of x_{α^c},

$$u_\alpha(\cdot, x_{\alpha^c}) = \mathcal{I}_{\mathcal{H}_\alpha} u(\cdot, x_{\alpha^c})$$

where $\mathcal{I}_{\mathcal{H}_\alpha}$ is some sample-based projection (e.g., interpolation, least-squares projection) onto a subspace $\mathcal{H}_\alpha$.

If the projection $\mathcal{I}_{\mathcal{H}_\alpha}$ is based on a set of M_α samples of X_α, obtaining U_α requires the evaluation of u at the $M_\alpha \times N_\alpha$ points

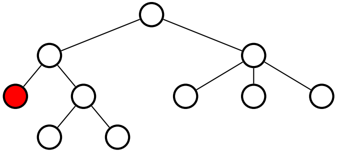
$$\{(x^i_\alpha, x^j_{\alpha^c}) : 1 \leq i \leq M_\alpha, 1 \leq j \leq N_\alpha\}.$$

Empirical principal component analysis for tree-based format

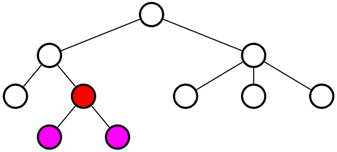
Given a tree T, the subspaces $U_\alpha \subset \mathcal{H}_\alpha$ are determined from the leaves to the root, and the spaces $\mathcal{H}_\alpha$ are chosen as follows.

- for $S(\alpha) = \emptyset$ (leaf node), $\mathcal{H}_\alpha$ is a given approximation space (e.g., polynomials, wavelets, kernel functions, perceptrons...)

$$\mathcal{H}_\alpha = \mathrm{span}\{\phi_\lambda^\alpha : \lambda \in I^\alpha\}$$



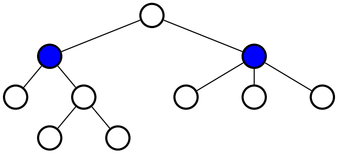
- for $S(\alpha) \neq \emptyset$ (interior node), $\mathcal{H}_\alpha = \bigotimes_{\beta \in S(\alpha)} U_\beta$.



Empirical principal component analysis for tree-based format

We finally obtain an approximation $u^\star$ of u by a sample-based projection (e.g., interpolation or least-squares projection) onto the tensor space $\mathcal{H}_D = \bigotimes_{\alpha \in S(D)} U_\alpha$

$$u^\star = \mathcal{I}_{\mathcal{H}_D} u$$



Empirical principal component analysis for tree-based format

Theorem (Fixed precision, using interpolation)

Let $\epsilon, \tilde{\epsilon} \geq 0$. If the subspaces U_α are determined such that

$$\|\mathcal{P}_{U_\alpha} u_\alpha - u_\alpha\| \leq \frac{\epsilon}{\sqrt{\# T}} \|u_\alpha\|$$

and if the approximation spaces $\mathcal{H}_\nu$, $1 \leq \nu \leq d$, are such that

$$\|\mathcal{P}_{\mathcal{H}_\nu} u - u\| \leq \tilde{\epsilon} \|u\|,$$

then we obtain an approximation $u^\star$ such that

$$\|u^\star - u\|^2 \leq (\Lambda^2 \epsilon^2 + \tilde{\Lambda}^2 \tilde{\epsilon}^2) \|u\|^2$$

with Λ and $\tilde{\Lambda}$ depending on the properties of the interpolation operators.

Illustration for approximation: Borehole function

The Borehole function models water flow through a borehole:

$$u(X) = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)\left(1 + \frac{2LT_u}{\ln(r/r_w)r_w^2 K_w} + \frac{T_u}{T_l}\right)}, \quad X = (r_w, \log(r), T_u, H_u, T_l, H_l, L, K_w)$$

r_w	radius of borehole (m)	$N(\mu = 0.10, \sigma = 0.0161812)$
r	radius of influence (m)	$LN(\mu = 7.71, \sigma = 1.0056)$
T_u	transmissivity of upper aquifer (m^2/yr)	$U(63070, 115600)$
H_u	potentiometric head of upper aquifer (m)	$U(990, 1110)$
T_l	transmissivity of lower aquifer (m^2/yr)	$U(63.1, 116)$
H_l	potentiometric head of lower aquifer (m)	$U(700, 820)$
L	length of borehole (m)	$U(1120, 1680)$
K_w	hydraulic conductivity of borehole (m/yr)	$U(9855, 12045)$

Approximation in the tree-based format $\mathcal{T}_r^T(\mathcal{H})$ with a linear tree

$$T = \{\{1\}, \dots, \{d\}, \{1,2\}, \dots, \{1, \dots, d-1\}, D\}$$

and polynomial approximation spaces $\mathcal{H}_\nu$, $1 \leq \nu \leq d$.

Illustration for approximation: Borehole function

Table: Approximation with prescribed precision ϵ, adaptive degree $p(\epsilon) = \log_{10}(\epsilon^{-1})$, and $N_\alpha = \dim(V_\alpha)$. Confidence intervals for relative error $\varepsilon(u^\star)$, storage complexity S and number of evaluations M for different ϵ, and average ranks. Projections based on empirical interpolation

ϵ	$\varepsilon(u^\star)$	N	S	$[r_{\{1\}}, \dots, r_{\{d\}}, r_{\{1,2\}}, \dots, r_{\{1,\dots,d-1\}}$
10^{-1}	$[1.8; 2.7] \times 10^{-1}$	$[39, 39]$	$[23, 23]$	$[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$
10^{-2}	$[0.3; 4.0] \times 10^{-2}$	$[88, 100]$	$[41, 46]$	$[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1]$
10^{-3}	$[0.8; 1.9] \times 10^{-3}$	$[159, 186]$	$[61, 78]$	$[2, 1, 1, 2, 2, 1, 1, 1, 1, 2, 2, 2, 1, 1]$
10^{-4}	$[2.5; 5.6] \times 10^{-5}$	$[328, 328]$	$[141, 141]$	$[2, 2, 2, 3, 3, 2, 2, 2, 1, 2, 2, 2, 2, 2]$
10^{-5}	$[0.6; 1.6] \times 10^{-5}$	$[444, 472]$	$[166, 178]$	$[2, 2, 2, 4, 4, 2, 2, 2, 1, 2, 2, 2, 2, 2]$
10^{-6}	$[3.1; 5.7] \times 10^{-6}$	$[596, 664]$	$[204, 241]$	$[3, 2, 2, 4, 5, 3, 2, 2, 2, 2, 2, 2, 2, 2]$
10^{-7}	$[1.0; 6.3] \times 10^{-7}$	$[1042, 1267]$	$[374, 429]$	$[4, 3, 4, 6, 5, 3, 3, 3, 2, 2, 3, 2, 2, 2]$
10^{-8}	$[1.1; 7.1] \times 10^{-8}$	$[1567, 1567]$	$[512, 512]$	$[4, 3, 4, 7, 6, 3, 3, 3, 2, 2, 3, 2, 3, 3]$
10^{-9}	$[0.2; 4.9] \times 10^{-8}$	$[1719, 1854]$	$[534, 560]$	$[4, 4, 4, 8, 6, 3, 3, 3, 2, 2, 3, 2, 3, 3]$
10^{-10}	$[0.3; 1.9] \times 10^{-9}$	$[2482, 2828]$	$[774, 838]$	$[5, 4, 6, 10, 7, 4, 3, 3, 2, 2, 3, 2, 3, 3]$

Approximation of a function using tensorization

Consider a function $u : [0,1) \to 1$ identified with the multivariate function

$$u(x) = v(i_0, \dots, i_{d-1}, y), \quad x = 2^{-d}(y + \sum_{k=0}^{d-1} i_k 2^k)$$

with $y \in [0,1)$ and $i_0, \dots, i_{d-1} \in \{0,1\}$.

The function v is approximated in the tree-based format $\mathcal{T}_r^T(\mathcal{H})$ with a linear tree

$$T = \{\{1\}, \dots, \{d\}, \{1,2\}, \dots, \{1, \dots, d-1\}, D\}$$

and $\mathcal{H} = \mathbb{R}^2 \otimes \dots \otimes \mathbb{R}^2 \otimes \mathbb{P}_0$. This results in a piecewise constant approximation of u on a uniform partition of $[0,1]$ with 2^d intervals.

Approximation of a function using tensorization

Table: $u(t) = \sqrt{t}$, $d = 40$. Approximation in tensor train format with prescribed ϵ, $N_\alpha = \dim(\mathcal{H}_\alpha)$. Confidence intervals for relative L^2-error $\varepsilon(u^\star)$, number of evaluations M, storage complexity S and maximal rank for different ϵ.

ϵ	$\varepsilon(u^\star)$	M	S	$\max_\alpha r_\alpha$
10^{-1}	$[9.3\,10^{-3}; 5.5\,10^{-2}]$	$[182, 230]$	$[90, 114]$	$[2, 2]$
10^{-2}	$[3.7\,10^{-3}; 8.6\,10^{-3}]$	$[314, 350]$	$[156, 172]$	$[2, 3]$
10^{-3}	$[5.4\,10^{-4}; 9.2\,10^{-4}]$	$[514, 606]$	$[252, 300]$	$[3, 3]$
10^{-4}	$[1.3\,10^{-4}; 3.3\,10^{-3}]$	$[838, 962]$	$[414, 474]$	$[4, 4]$
10^{-5}	$[1.8\,10^{-5}; 8.2\,10^{-4}]$	$[1270, 1398]$	$[626, 692]$	$[4, 5]$
10^{-6}	$[1.3\,10^{-6}; 6.3\,10^{-5}]$	$[1900, 2036]$	$[938, 1014]$	$[5, 5]$
10^{-7}	$[4.9\,10^{-7}; 1.3\,10^{-6}]$	$[2444, 2718]$	$[1218, 1344]$	$[5, 6]$
10^{-8}	$[1.0\,10^{-7}; 1.2\,10^{-6}]$	$[3304, 3468]$	$[1642, 1722]$	$[6, 6]$
10^{-9}	$[2.2\,10^{-8}; 1.3\,10^{-7}]$	$[4116, 4328]$	$[2046, 2144]$	$[7, 7]$
10^{-10}	$[8.6\,10^{-10}; 6.7\,10^{-8}]$	$[5024, 5136]$	$[2490, 2552]$	$[7, 7]$

Concluding remarks

- A fundamental problem would be to characterize approximation classes $\mathcal{A}^\gamma$ of tree tensor networks, those functions for which tree tensor networks give a certain performance
$$\inf_{v \in \mathcal{T}_r^T(\mathcal{H})} \mathcal{R}(\hat{u}) - \mathcal{R}(u) \lesssim \gamma(n)^{-1}$$
for some growth function γ and n a measure of complexity of $\mathcal{T}_r^T(\mathcal{H})$.

- For an approximation class $\mathcal{A}^\gamma$, we would like to devise (black box) algorithms that select a model class $\mathcal{T}_r^T(\mathcal{H})$ and provide an approximation $\hat{u} \in \mathcal{T}_r^T(\mathcal{H})$ such that (possibly in expectation or with a certain probability)
$$\mathcal{R}(\hat{u}) - \mathcal{R}(u) \lesssim \gamma(n)^{-1},$$
or γ replaced by another growth function.

References I

W. Hackbusch.
Tensor spaces and numerical tensor calculus, volume 42 of *Springer series in computational mathematics*.
Springer, Heidelberg, 2012.

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.

A. Nouy.
Low-Rank Tensor Methods for Model Order Reduction, pages 857–882.
Springer International Publishing, Cham, 2017.

M. Bachmayr, R. Schneider, and A. Uschmajew.
Tensor networks and hierarchical tensors for the solution of high-dimensional partial differential equations.
Foundations of Computational Mathematics, pages 1–50, 2016.

B. Khoromskij.
Tensors-structured numerical methods in scientific computing: Survey on recent advances.
Chemometrics and Intelligent Laboratory Systems, 110(1):1 – 19, 2012.

E. Grelier, A. Nouy, M. Chevreuil.
Learning with tree-based tensor formats.
Arxiv eprints, Nov. 2018.

References II

N. Cohen, O. Sharir, and A. Shashua.
On the expressive power of deep learning: A tensor analysis.
In *Conference on Learning Theory*, pages 698–728, 2016.

V. Khrulkov, A. Novikov, and I. Oseledets.
Expressive power of recurrent neural networks.
In *International Conference on Learning Representations*, 2018.

V. Kazeev and C. Schwab.
Approximation of singularities by quantized-tensor fem.
PAMM, 15(1):743–746, 2015.

V. Kazeev, I. Oseledets, M. Rakhuba, and C. Schwab.
Qtt-finite-element approximation for multiscale problems i: model problems in one dimension.
Advances in Computational Mathematics, 43(2):411–442, Apr 2017.

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.

R. Schneider and A. Uschmajew.
Approximation rates for the hierarchical tensor format in periodic sobolev spaces.
Journal of Complexity, 30(2):56 – 71, 2014.
Dagstuhl 2012.