**ETICS research school**
3-7 oct. 2022

# High-Dimensional Approximation

## Part 2: High-dimensional approximation tools

**Anthony Nouy**

Centrale Nantes, Nantes Université, Laboratoire de Mathématiques Jean Leray

# Approximation tools for high-dimensional approximation

We here present classical approximation tools (model classes) for the approximation of multivariate functions

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

# Outline

1. Overview of classical approximation tools

2. Approximation theory of (deep) neural networks

3. Approximation theory of tree tensor networks

# Expansions on tensor product bases

- Polynomial models

$$\sum_{\alpha \in \Lambda} a_\alpha x^\alpha, \quad x^\alpha = x_1^{\alpha_1} \dots x_d^{\alpha_d}$$

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

# Expansions on tensor product bases

- Polynomial models
$$\sum_{\alpha \in \Lambda} a_\alpha x^\alpha, \quad x^\alpha = x_1^{\alpha_1} \dots x_d^{\alpha_d}$$
where $\Lambda \subset \mathbb{N}^d$ is a set of multi-indices, either fixed (linear approximation) or free (nonlinear approximation).

- More general expansions on tensorized bases
$$\sum_{\alpha \in \Lambda} a_\alpha \psi_\alpha(x), \quad \psi_\alpha(x) = \psi_{\alpha_1}(x_1) \dots \psi_{\alpha_d}(x_d),$$
e.g. tensor product splines, wavelets...

# Additive and multiplicative models

- Additive models

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

or more generally

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

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

# Additive and multiplicative models

- Additive models

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

  or more generally

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

  where $\Lambda \subset 2^{\{1,\ldots,d\}}$ 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

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

  where $\Lambda \subset 2^{\{1,\ldots,d\}}$ is either a fixed or a free parameter.

# Separation of variables and tensor networks

- Sum of multiplicative models (canonical tensor format)

$$\sum_{k=1}^{r} v^{(1)}(x_1, k) \dots v^{(d)}(x_d, k)$$

that is a $r$-term approximation from the dictionary of separated functions.

# Separation of variables and tensor networks

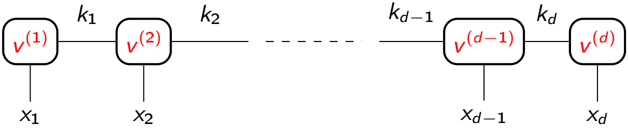
- Sum of multiplicative models (canonical tensor format)

$$\sum_{k=1}^{r} v^{(1)}(x_1, k) \dots v^{(d)}(x_d, k)$$

that is a $r$-term approximation from the dictionary of separated functions.

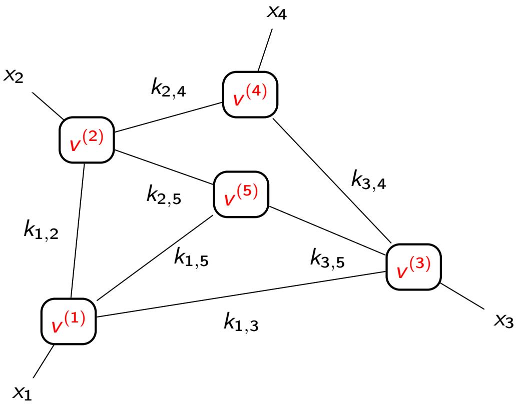
- Tensor train (Matrix Product State)

$$v(x) = \sum_{k_1=1}^{r_1} \dots \sum_{k_{d-1}=1}^{r_{d-1}} v^{(1)}(x_1, k_1) v^{(2)}(k_1, x_2, k_2) \dots v^{(d)}(k_{d-1}, x_d).$$




It is a particular case of tensor networks.

- Tensor networks associated with general graphs



# Composition of functions

$$f(g(x))$$

with $g : \mathbb{R}^d \to \mathbb{R}^m$ and $f : \mathbb{R}^m \to \mathbb{R}$.

$g$ can be seen as a map that extracts $m$ features $g(x)$ (new variables) from an input $x$, that can be fixed (application-dependent) or free.

For linear maps $g(x) = Ax$, this corresponds to ridge approximation

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

# Composition of functions

$$f(g(x))$$

with $g : \mathbb{R}^d \to \mathbb{R}^m$ and $f : \mathbb{R}^m \to \mathbb{R}$.

$g$ can be seen as a map that extracts $m$ features $g(x)$ (new variables) from an input $x$, that can be fixed (application-dependent) or free.

For linear maps $g(x) = Ax$, this corresponds to ridge approximation

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

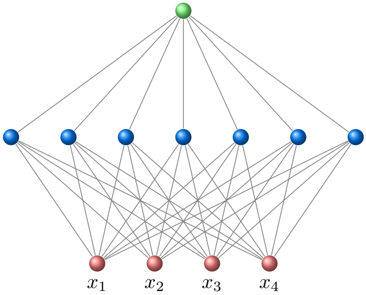
Different regimes

- small $m$, $g$ performs a dimension reduction and $f$ is a low-dimensional function.
- large $m$, $g$ extracts many features and $f$ is expected to be simple, e.g. linear or additive.

A shallow neural network (with one hidden layer of width $m$) is a ridge function

$$a^T \sigma(Ax + b) = \sum_{i=1}^{m} a_i \sigma(\sum_{j=1}^{d} A_{ij} x_j + b_i)$$

where $\sigma$ is a given function (activation function).



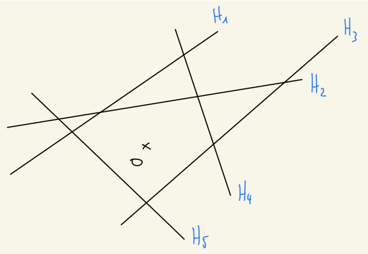
# Neural networks

Classical piecewise polynomial activation functions

- ReLU function $\sigma(t) = \langle t \rangle_+ = \max\{0, t\}$
- RePU(p) function $\sigma(t) = \langle t \rangle_+^p = \max\{0, t\}^p$

ReLU and RePU networks produce a piecewise polynomial approximation (spline) on a free partition of $\mathbb{R}^d$ determined by $m$ hyperplanes

$$H_i = \{x : {w_i}^T x + b_i = 0\}, \quad w_i = (A_{ij})_{j=1}^d \in \mathbb{R}^d$$



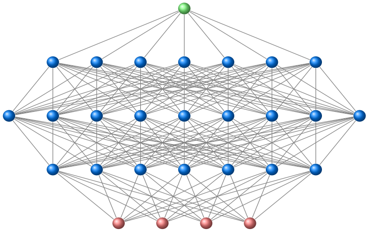
# Deep neural networks

$$T_L \circ \sigma \circ T_{L-1} \circ \ldots \circ T_1 \circ \sigma \circ T_0(x)$$

with $T_\ell : \mathbb{R}^{m_\ell} \to \mathbb{R}^{m_{\ell+1}}$ an affine linear map

$$T_\ell(x) = A_\ell x + b_\ell$$

and $(m_1, \ldots, m_L) \in \mathbb{N}^L$ with $m_0 = d$, $m_{L+1} = 1$.



For ReLU or RePU(p) activation function $\sigma$, the approximation is a piecewise polynomial on a free partition with a number of domains growing exponentially with depth $L$.

## Outline

Different approximation tools $(X_n)_{n\geq 1}$ can be defined depending on which parameters are free (possible architectures) and how complexity is measured.

Letting $\Phi_{L,m}$ be the class of neural networks with depth $L$ and widths $m = (m_1, \dots, m_L)$, we define

$$X_n = \{v \in \Phi_{L,m} : L \in \mathcal{L}, m \in \mathcal{M}_L, compl(v) \leq n\}$$

where *compl* is a complexity measure, $\mathcal{L} \subset \mathbb{N}$ is the set of possible depths and $\mathcal{M}_L \subset \mathbb{N}^L$ the set of possible widths.

Two typical classes of architectures

- Fixed depth $L$ and free width:

$$\mathcal{L} = \{L\}, \quad \mathcal{M}_L = \{(W, \dots, W) : W \in \mathbb{N}\}$$

- Free depth and fixed width $W$:

$$\mathcal{L} = \mathbb{N}, \quad \mathcal{M}_L = \{(W, \dots, W)\}$$

# Approximation tools based on neural networks

For a function $v$ in the class $\Phi_{L,m}$ of neural networks with depth $L$ and widths $m = (m_1, \dots, m_L)$, different measures of complexity:

- number of parameters (fully connected networks)

$$compl_F(v) = \sum_{\ell=0}^{L} m_\ell m_{\ell+1} + m_{\ell+1} \sim W^2 L \text{ for } m_\ell \sim W$$

- number of non-zero parameters (sparsely connected networks)

$$compl_S(v) = \sum_{\ell=0}^{L} \|A_\ell\|_0 + \|b_\ell\|_0$$

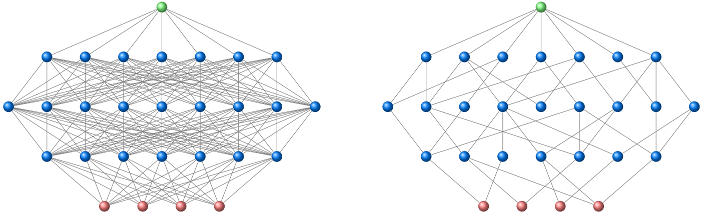


Figure: Fully connected network (left) and Sparsely connected network (right).

Structured sparsity can be imposed (convolutional NN, recurrent NN...) or sparsity pattern can be considered as a free parameter (a challenge on the algorithmic side).

# Deep neural networks approximation theory

Many recent results on the expressivity of deep neural networks for various model classes.

- Approximation classes of deep neural networks (free depth and fixed width) are larger than those of shallow networks (fixed depth and free width) [DeVore et al 2020].

# Deep neural networks approximation theory

Many recent results on the expressivity of deep neural networks for various model classes.

- Approximation classes of deep neural networks (free depth and fixed width) are larger than those of shallow networks (fixed depth and free width) [DeVore et al 2020].
- Deep neural networks are (almost) as expressive as many classical approximation tools (polynomials, splines, B-splines...).

# Deep neural networks approximation theory

Many recent results on the expressivity of deep neural networks for various model classes.

- Approximation classes of deep neural networks (free depth and fixed width) are larger than those of shallow networks (fixed depth and free width) [DeVore et al 2020].
- Deep neural networks are (almost) as expressive as many classical approximation tools (polynomials, splines, B-splines...).
- They achieve (near to) optimal performance for functions from classical smoothness classes (isotropic or anisotropic Sobolev, Besov, analytic functions...).

  For functions $u$ in $W^{s,\infty}((0,1)^d)$, ReLU networks achieve

  $$e_n(u)_{L^\infty} \lesssim n^{-d/s}$$

  with continuous parameter selection.

# Deep neural networks approximation theory

Many recent results on the expressivity of deep neural networks for various model classes.

- Approximation classes of deep neural networks (free depth and fixed width) are larger than those of shallow networks (fixed depth and free width) [DeVore et al 2020].
- Deep neural networks are (almost) as expressive as many classical approximation tools (polynomials, splines, B-splines...).
- They achieve (near to) optimal performance for functions from classical smoothness classes (isotropic or anisotropic Sobolev, Besov, analytic functions...).

  For functions $u$ in $W^{s,\infty}((0,1)^d)$, ReLU networks achieve

  $$e_n(u)_{L^\infty} \lesssim n^{-d/s}$$

  with continuous parameter selection.
- Approximation classes of deep ReLU networks are not embedded in standard smoothness classes [Gribonval et al 2021]

# Deep neural networks approximation theory

Many recent results on the expressivity of deep neural networks for various model classes.

- Approximation classes of deep neural networks (free depth and fixed width) are larger than those of shallow networks (fixed depth and free width) [DeVore et al 2020].
- Deep neural networks are (almost) as expressive as many classical approximation tools (polynomials, splines, B-splines...).
- They achieve (near to) optimal performance for functions from classical smoothness classes (isotropic or anisotropic Sobolev, Besov, analytic functions...).

  For functions $u$ in $W^{s,\infty}((0,1)^d)$, ReLU networks achieve

  $$e_n(u)_{L^\infty} \lesssim n^{-d/s}$$

  with continuous parameter selection.
- Approximation classes of deep ReLU networks are not embedded in standard smoothness classes [Gribonval et al 2021]
- They approximate efficiently functions beyond smoothness classes (discontinuous functions, fractals, refinable functions...)

A few surprises

- For functions $u$ in the unit ball $K$ of $W^{s,\infty}((0,1)^d)$, ReLU networks with free depth can achieve
$$e_n(u)_{L^\infty} \lesssim n^{-p} \quad \text{for arbitrary } p \leq 2s/d.$$
However, since the manifold width $\delta_n(K)_{L^\infty} \gtrsim n^{-s/d}$, a rate $p > s/d$ can be achieved only with discontinuous parameter selection. Also, it requires an encoding of parameters with more than $O(\log_2(\epsilon^{-1}))$ bits to achieve accuracy $\epsilon$.

# Deep neural networks approximation theory

A few surprises

- For functions $u$ in the unit ball $K$ of $W^{s,\infty}((0,1)^d)$, ReLU networks with free depth can achieve
$$e_n(u)_{L^\infty} \lesssim n^{-p} \quad \text{for arbitrary } p \leq 2s/d.$$
However, since the manifold width $\delta_n(K)_{L^\infty} \gtrsim n^{-s/d}$, a rate $p > s/d$ can be achieved only with discontinuous parameter selection. Also, it requires an encoding of parameters with more than $O(\log_2(\epsilon^{-1}))$ bits to achieve accuracy $\epsilon$.
- Approximation classes of deep networks contain functions that could in principle be approximated without the curse of dimensionality but require in practice an exponential quantity of information. That is the theory to practice gap [Grohs and Voigtlaender 2021].

# Deep neural networks approximation theory

A few surprises

- For functions $u$ in the unit ball $K$ of $W^{s,\infty}((0,1)^d)$, ReLU networks with free depth can achieve
$$e_n(u)_{L^\infty} \lesssim n^{-p} \quad \text{for arbitrary } p \leq 2s/d.$$
However, since the manifold width $\delta_n(K)_{L^\infty} \gtrsim n^{-s/d}$, a rate $p > s/d$ can be achieved only with discontinuous parameter selection. Also, it requires an encoding of parameters with more than $O(\log_2(\epsilon^{-1}))$ bits to achieve accuracy $\epsilon$.
- Approximation classes of deep networks contain functions that could in principle be approximated without the curse of dimensionality but require in practice an exponential quantity of information. That is the theory to practice gap [Grohs and Voigtlaender 2021].

Open problems

- Characterize the functions that can be approximated stably with deep networks.
- Characterize functions that can be estimated with partial information and near optimal performance.
- Provide algorithms that achieve near to optimal performance.

## Outline

# Tensor networks

Tensor networks are prominent tools for the representation or approximation of multivariate functions or multidimensional arrays.

- A long history in quantum physics. A zoo of tools exploiting separation of variables (MPS, PEPS, MERA...)
- Tree tensor networks (Hierarchical Tucker tensors) appeared independently in numerical analysis and numerical linear algebra, as an extension of low-rank decompositions to high-order tensors [Hackbusch and Kuhn, Grasedyck, Oseledets and Tyrtyshnikov].
- Growing use in statistics, data science and probabilistic modelling.

# Tensor product of functions

Let $V_\nu \subset \mathbb{R}^{\mathcal{X}_\nu}$ be a space of functions defined on $\mathcal{X}_\nu$.

$\mathcal{X}_\nu$ can be (a subset of) $\mathbb{N}$, $\mathbb{R}$, $\mathbb{C}$, or a set of vectors, sequences, graphs, images...

The tensor product of functions $v^{(\nu)} \in V_\nu$, denoted

$$v = v^{(1)} \otimes \ldots \otimes v^{(d)},$$

is a multivariate function defined on $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_d$ and such that

$$v(x_1, \ldots, x_d) = v^{(1)}(x_1) \ldots v^{(d)}(x_d)$$

# Tensor product of functions

The algebraic tensor product of spaces $V_\nu$ is defined as

$$V_1 \otimes \ldots \otimes V_d = \mathrm{span}\{v^{(1)} \otimes \ldots \otimes v^{(d)} : v^{(\nu)} \in V_\nu, 1 \leq \nu \leq d\}$$

which is the space of multivariate functions $v$ which can be written as a finite linear combination of elementary (separated functions), i.e.

$$v(x_1, \ldots, x_d) = \sum_{k=1}^{r} v_k^{(1)}(x_1) \ldots v_k^{(d)}(x_d).$$

# Rank of multivariate functions and canonical format

The canonical rank of a multivariate function $f(x_1, \ldots, x_d)$ is the minimal integer such that $f$ has a representation

$$f(x) = \sum_{k=1}^{r} v_k^1(x_1) \ldots v_k^d(x_d)$$

Given a finite-dimensional tensor space $V = V_1 \otimes \ldots \otimes V_d$ of multivariate functions we define a canonical tensor format in $V$ as a set of functions

$$\mathcal{R}_r(V) = \{f \in V : \mathrm{rank}(f) \leq r\}$$

# Rank of multivariate functions and canonical format

The canonical rank of a multivariate function $f(x_1, \dots, x_d)$ is the minimal integer such that $f$ has a representation

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

Given a finite-dimensional tensor space $V = V_1 \otimes \dots \otimes V_d$ of multivariate functions we define a canonical tensor format in $V$ as a set of functions

$$\mathcal{R}_r(V) = \{f \in V : \text{rank}(f) \leq r\}$$

From the practical point of view, it is not a nice format. In particular, $\mathcal{R}_r(V)$ is not closed for $d \geq 3$ and $r \geq 2$.

For any continuous parametrization $\mathcal{R}_r(V) = \{v = R(p) : p \in P\}$, and for any tensor of $v \in \overline{\mathcal{R}_r(V)} \setminus \mathcal{R}_r(V)$ of border rank $r$, the quantity

$$\delta(v, \epsilon) = \inf\{\|p\| : \|v - R(p)\| < \epsilon\}$$

diverges as $\epsilon \to 0$ [Hackbusch 2021].

A multivariate function $f(x_1, \ldots, x_d)$, for any set $\alpha \subset \{1, \ldots, d\}$, can be identified with a bivariate function $f(x_\alpha, x_{\alpha^c})$ of two complementary subsets of variables.

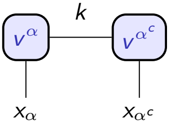
The rank of the bivariate function $f(x_\alpha, x_{\alpha^c})$ is the $\alpha$-rank of $f$, denoted $\mathrm{rank}_\alpha(f)$.

A function with $\alpha$-rank bounded by $r_\alpha$ admits a representation

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

or using tensor diagram notations

$f(x) =$




where a connection between two tensors represents a contraction along one mode of each tensor.

# $\alpha$-rank

## Example

- $u(x) = u^1(x_1)\dots u^d(x_d)$ can be written $u(x) = u^\alpha(x_\alpha)u^{\alpha^c}(x_{\alpha^c})$, with $u^\alpha(x_\alpha) = \prod_{\nu\in\alpha} u^\nu(x_\nu)$. Therefore, for any $\alpha$, $\text{rank}_\alpha(u) = 1$.

- $u(x) = \sum_{k=1}^r u_k^1(x_1)\dots u_k^d(x_d)$ can be written $\sum_{k=1}^r u_k^\alpha(x_\alpha)u_k^{\alpha^c}(x_{\alpha^c})$ with $u_k^\alpha(x_\alpha) = \prod_{\nu\in\alpha} u_k^\nu(x_\nu)$. Therefore, for any $\alpha$, $\text{rank}_\alpha(u) \le r$, with equality if the functions $\{u_k^\alpha(x_\alpha)\}$ and the functions $\{u_k^{\alpha^c}(x_{\alpha^c})\}$ are linearity independent.

  We deduce the following relation between $\alpha$-ranks and canonical rank:

  $$\text{rank}_\alpha(u) \le \text{rank}(u), \quad \text{for any } \alpha.$$

- $u(x) = u^1(x_1) + \dots + u^d(x_d)$ can be written $u(x) = u^\alpha(x_\alpha) + u^{\alpha^c}(x_{\alpha^c})$, with $u^\alpha(x_\alpha) = \sum_{\nu\in\alpha} u^\nu(x_\nu)$. Therefore, $\text{rank}_\alpha(u) \le 2$.

## Low-rank tensor format

Given

- a finite-dimensional tensor space $V = V_1 \otimes \ldots \otimes V_d$ of multivariate functions
- a collection $T$ of subsets in $\{1, \ldots, d\}$,
- a tuple of ranks $r = (r_\alpha)_{\alpha \in T}$,

we define a low-rank tensor format in $V$ as a set of functions

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

# Low-rank tensor format

Given

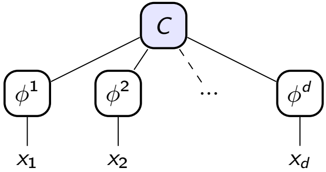
- a finite-dimensional tensor space $V = V_1 \otimes \ldots \otimes V_d$ of multivariate functions
- a collection $T$ of subsets in $\{1, \ldots, d\}$,
- a tuple of ranks $r = (r_\alpha)_{\alpha \in T}$,

we define a low-rank tensor format in $V$ as a set of functions

$$\mathcal{T}_r^T(V) = \{f \in V : \mathrm{rank}_\alpha(f) \le r_\alpha, \alpha \in T\}$$

with representation

$$f(x) = \sum_{i_1 \in I_1} \ldots \sum_{i_d \in I_d} C(i_1, \ldots, i_d)\phi^1(x_1)_{i_1} \ldots \phi^d(x_d)_{i_d} =$$




where $\phi^\nu$ is a feature map associated with $V^\nu$ and $C \in \mathbb{R}^{I_1 \times \ldots \times I_d}$ is a rank-structured algebraic tensor.

# Tensor train format

With

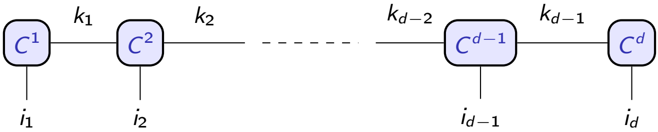
$$T = \{\{1\}, \{1,2\}, ..., \{1,\ldots,d\}\},$$

$\mathcal{T}_r^T(V)$ coincides with the tensor train format.

A function $f$ in $\mathcal{T}_r^T(V)$ has coefficients

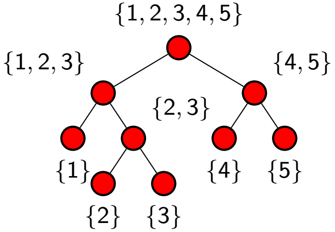
$$C(i_1,\ldots,i_d) = \sum_{k_1=1}^{r_1} \cdots \sum_{k_{d-1}=1}^{r_{d-1}} C^1(i_1,k_1)C^2(k_1,i_2,k_2)\ldots C^d(k_{d-1},i_d).$$

$=$

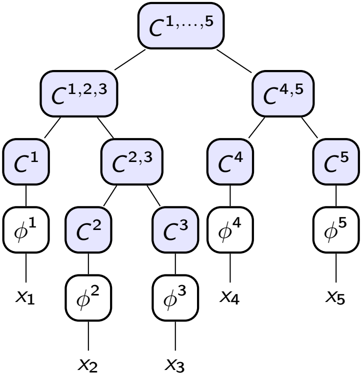


# Hierarchical Tucker format (Tree tensor networks)

If $T$ is a dimension partition tree, $\mathcal{T}_r^T(V)$ is a tree-based (or hierarchical) tensor format and a function in $\mathcal{T}_r^T(V)$ admits a multilinear parametrization with a collection of parameters $\{C^\alpha : \alpha \in T\}$ forming a tree tensor network.




Dimension tree $T$




Tree tensor network

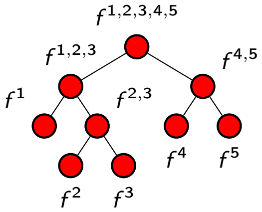
# Tree tensor networks as a compositional function network

By identifying a tensor $C^{(\alpha)} \in \mathbb{R}^{n_1 \times \ldots \times n_s \times r_\alpha}$ 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 $\mathcal{T}_r^T(V)$ admits a representation as a tree-structured composition of multilinear functions $\{f^{(\alpha)}\}_{\alpha \in T}$, e.g.

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



A multilinear map $f^\alpha$ can also be written

$$f^\alpha(z_1, \dots, z_s) = A^\alpha \sigma(z_1, \dots, z_d), \quad z_k \in \mathbb{R}^{n_k},$$

with a matrix

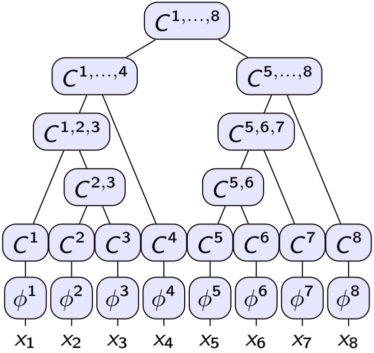
$$A^\alpha \in \mathbb{R}^{r_\alpha \times N}, \quad N = n_1 \dots n_s$$

and a fixed multilinear function

$$\sigma(z_1, \dots, z_s) = vec(z_1 \otimes \dots \otimes z_s) \in \mathbb{R}^N$$

# Tree tensor networks as feed-forward neural networks

It corresponds to a sum-product feed forward neural network with a sparse architecture (given by $T$), a number of hidden layers equal to $depth(T)+1$ (including a featuring layer), and width at level $\ell$ related to the $\alpha$-ranks of the nodes $\alpha$ of level $\ell$.



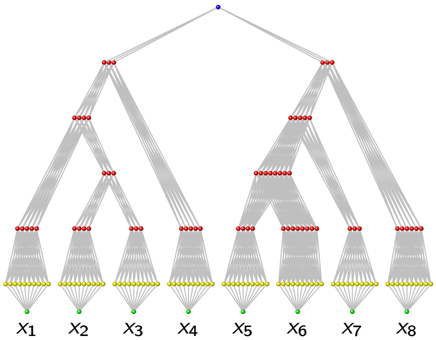




Figure: Tree tensor network and corresponding feed-forward sum-product neural network with 10 features per variable $x_\nu$ (right)

# Outline

# Approximation tools based on tree tensor networks

For the approximation of a function, a first approach is to introduce subspaces $V^\nu_{N_\nu}$ of finite dimension (e.g. polynomials, splines, wavelets, RKHS...) and consider tree tensor networks $f \in \mathcal{T}_r^T(V_N)$ where

$$V_N = V^1_{N_1} \otimes \ldots \otimes V^d_{N_d},$$

with variable $N$ and $r$.

Spaces $V^\nu_{N_\nu}$ have to be well chosen, e.g. polynomials for analytic functions, splines with a degree adapted to the regularity of the function...

An approximation tool $\Phi = (\Phi_n)_{n\in\mathbb{N}}$ is then defined by

$$\Phi_n = \{f \in \mathcal{T}_r^T(V_N) : N \in \mathbb{N}^d, r \in \mathbb{N}^T, \mathit{compl}(f) \leq n\}.$$

The dimensions $N$ and the ranks $r$ are free parameters, and $\mathit{compl}(\cdot)$ is some complexity measure.

An approximation tool $\Phi = (\Phi_n)_{n\in\mathbb{N}}$ is then defined by

$$\Phi_n = \{f \in \mathcal{T}_r^T(V_N) : N \in \mathbb{N}^d, r \in \mathbb{N}^T, \mathit{compl}(f) \leq n\}.$$

The dimensions $N$ and the ranks $r$ are free parameters, and $\mathit{compl}(\cdot)$ is some complexity measure.

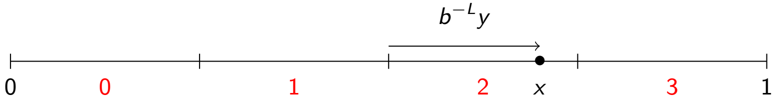
An alternative approach is to rely on tensorization of functions (specific featuring step).

# Tensorization of univariate functions

Consider a function $f \in \mathbb{R}^{[0,1)}$ defined on the interval $[0,1)$.

- For $b, L \in \mathbb{N}$, we subdivide uniformly the interval $[0,1)$ into $b^L$ intervals. Any $x \in [0,1)$ can be written

$$x = b^{-L}(i + y), \quad i \in \{0, \ldots, b^L - 1\}, \quad y \in [0,1).$$

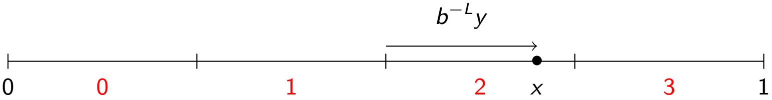


# Tensorization of univariate functions

Consider a function $f \in \mathbb{R}^{[0,1)}$ defined on the interval $[0,1)$.

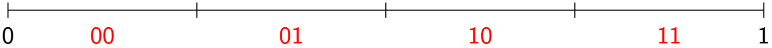
- For $b, L \in \mathbb{N}$, we subdivide uniformly the interval $[0,1)$ into $b^L$ intervals. Any $x \in [0,1)$ can be written

$$x = b^{-L}(i + y), \quad i \in \{0, \dots, b^L - 1\}, \quad y \in [0,1).$$




- The integer $i$ admits a representation in base $b$

$$i = \sum_{k=1}^{L} i_k b^{L-k} = [i_1 \dots i_L]_b, \quad i_k \in \{0, \dots, b-1\}$$

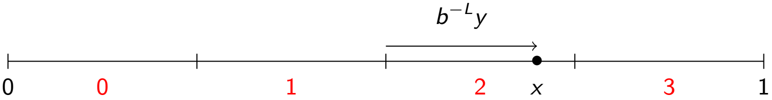


# Tensorization of univariate functions

Consider a function $f \in \mathbb{R}^{[0,1)}$ defined on the interval $[0,1)$.

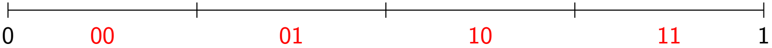
- For $b, L \in \mathbb{N}$, we subdivide uniformly the interval $[0,1)$ into $b^L$ intervals. Any $x \in [0,1)$ can be written

$$x = b^{-L}(i + y), \quad i \in \{0, \dots, b^L - 1\}, \quad y \in [0,1).$$




- The integer $i$ admits a representation in base $b$

$$i = \sum_{k=1}^{L} i_k b^{L-k} = [i_1 \dots i_L]_b, \quad i_k \in \{0, \dots, b-1\}$$

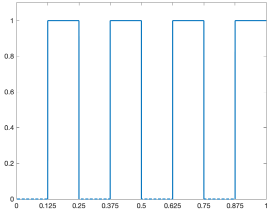



- $f$ is thus identified with a multivariate function (tensor of order $L+1$)

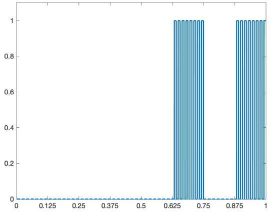
$$\boldsymbol{f} \in (\mathbb{R}^b)^{\otimes L} \otimes \mathbb{R}^{[0,1)} \quad \text{such that} \quad f(x) = \boldsymbol{f}(i_1, \dots, i_L, y)$$

# Tensorization of univariate functions

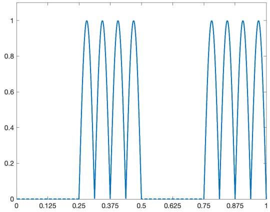
Digit $i_k(x)$ can be seen as a particular feature extracted from $x$.



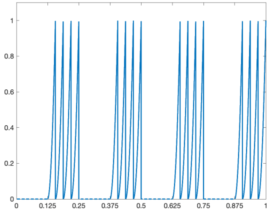

(a) $i_{\mathbf{3}}(x)$




(b) $i_{\mathbf{1}}(x)i_{\mathbf{3}}(x)i_{\mathbf{7}}(x)$




(c) $i_{\mathbf{2}}(x)sin(\pi y)$ $(L = 4)$




(d) $i_{\mathbf{3}}(x)y^{\mathbf{2}}$ $(L = 5)$

# Tensorization of multivariate functions

A function $f(x_1, \dots, x_d)$ defined on $[0,1)^d$ can be similarly identified with a tensor of order $(L+1)d$

$$\boldsymbol{f} \in (\mathbb{R}^b)^{\otimes Ld} \otimes (\mathbb{R}^{[0,1)})^{\otimes d}$$

such that

$$f(x_1, \dots, x_d) = \boldsymbol{f}(i_1^1, \dots, i_1^L, y_1, \dots, i_d^1, \dots, i_d^L, y_d)$$

$$\text{where} \quad x_\nu = b^{-L}(\sum_{k=1}^{L} i_\nu^k b^{L-k} + y_\nu) = [0.i_\nu^1 \dots i_\nu^L]_b + b^{-L} y_\nu$$

Digits $(i_1^1, \dots, i_1^L, \dots, i_d^1, \dots, i_d^L)$ encode a uniform partition of $[0,1)^d$ into $b^{dL}$ elements.

| | | | |
|---|---|---|---|
| 0011 | 0111 | 1011 | 1111 |
| 0010 | 0110 | 1010 | 1110 |
| 0001 | 0101 | 1001 | 1101 |
| 0000 | 0100 | 1000 | 1100 |

Figure: d=2, b=2, L=2

# Tensorization of multivariate functions

Using a different (resolution-wise) ordering of variables, the function can be identified with another tensor

$$f(x_1, \dots, x_d) = \boldsymbol{f}(i_1^1, \dots, i_d^1, \dots, i_1^L, \dots, i_d^L, y_1, \dots, y_d)$$

It corresponds to another encoding of the partition of $[0, 1)^d$ into $b^{dL}$ elements.

| | | | |
|---|---|---|---|
| 0101 | 0111 | 1101 | 1111 |
| 0100 | 0110 | 1100 | 1110 |
| 0001 | 0011 | 1001 | 1011 |
| 0000 | 0010 | 1000 | 1010 |

Figure: d=2, b=2, L=2

# Tensorization of multivariate functions

This particular re-parametrization is related to Morton space filling curve (or Z-order), which consists in mapping a point

$$([0.i_1^1 \dots i_1^L \dots]_2, \dots, [0.i_d^1 \dots i_d^L \dots]_2) \in [0,1]^d$$

to a real number

$$[0.i_1^1 \dots i_d^1 \dots i_1^L \dots i_d^L \dots]_2 \in [0,1]$$

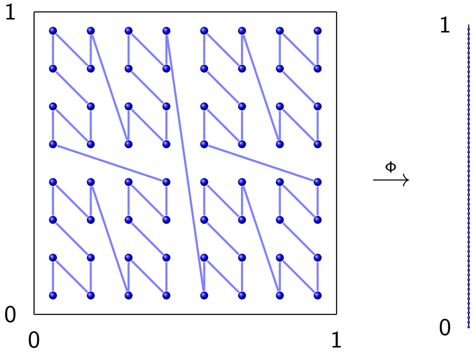
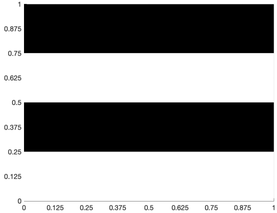


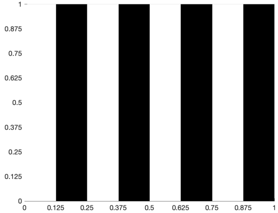

Figure: $b = 2$ and $L = 3$

# Tensorization of multivariate functions

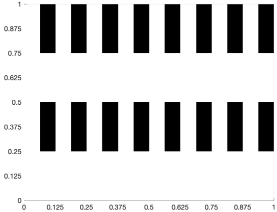
Digit $i_\nu^\ell(x)$ can be seen as a particular feature extracted from $x$.



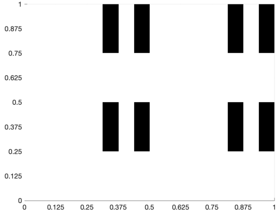

(a) $i_{\mathbf{1}}^{\mathbf{2}}(x)$




(b) $i_{\mathbf{2}}^{\mathbf{1}}(x)$




(c) $i_{\mathbf{1}}^{\mathbf{2}}(x) i_{\mathbf{3}}^{\mathbf{1}}(x)$




(d) $i_{\mathbf{1}}^{\mathbf{1}}(x) i_{\mathbf{1}}^{\mathbf{2}}(x) i_{\mathbf{3}}^{\mathbf{1}}(x)$

# Tensorization of multivariate functions

The map $T_{b,L}$ which associates to a function $f$ its tensorization $\boldsymbol{f}$ is a linear isometry from $L^p([0,1)^d)$ to $L^p(\{0,\dots,b-1\}^{Ld} \times [0,1)^d)$ for any $0 < p \leq \infty$.

We consider functions whose tensorization at resolution $L$ are in the tensor space

$$\boldsymbol{V}_L = (\mathbb{R}^b)^{\otimes Ld} \otimes S^{\otimes d}$$

with $S \subset \mathbb{R}^{[0,1)}$ some subspace of univariate functions, invariant through $b$-adic dilation.

We consider functions whose tensorization at resolution $L$ are in the tensor space

$$\boldsymbol{V}_L = (\mathbb{R}^b)^{\otimes Ld} \otimes S^{\otimes d}$$

with $S \subset \mathbb{R}^{[0,1)}$ some subspace of univariate functions, invariant through $b$-adic dilation.

If $S = \mathbb{P}_m$, $V_L = T_{b,L}^{-1}(\boldsymbol{V}_L)$ is identified with the space of multivariate splines of degree $m$ over a uniform partition with $b^{dL}$ elements, i.e.

$$V_L = V_{N_1}^1 \otimes \ldots \otimes V_{N_d}^d$$

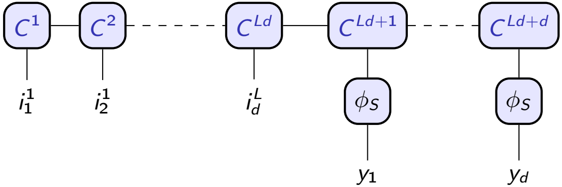
with $N_1 = ... = N_d = b^L$ and $V_{N_\nu}^\nu$ a space of univariate splines of degree $m$ over a uniform partition with $N_\nu = b^L$ intervals.

# Approximation tools based on tree tensor networks

Then as an approximation tool, we consider functions $f$ whose tensorization is a tensor network in $\mathcal{T}_r^{T_L}(\boldsymbol{V}_L)$, with $T_L$ a dimension tree over $\{1, \dots, Ld + d\}$.

Using the tensor train format, the corresponding function $f(x_1, \dots, x_d)$ has the representation

$f(x_1, \dots, x_d) =$




with $\phi_S$ the feature map associated with $S$. This is closely related to the quantized tensor train (QTT) format [Kazeev, Khoromskij, Oseledets, Schwab, ...]

Later on, we consider $S = \mathbb{P}_m$ and $\phi_S(y) = (1, y, ..., y^{m+1})$ or any other polynomial basis.

# Approximation tools based on tree tensor networks

An approximation tool $\Phi = (\Phi_n)_{n\in\mathbb{N}}$ is then defined by

$$\Phi_n = \{f \in \Phi_{L,T_L,r} : L \in \mathbb{N}_0, r \in \mathbb{N}^{T_L}, compl(f) \le n\}$$

with $\Phi_{L,T_L,r}$ the functions whose tensorization at resolution $L$ is in $\mathcal{T}_r^{T_L}(\boldsymbol{V}_L)$.

The resolution $L$ and ranks $r$ are free parameters, and $compl(\cdot)$ is some complexity measure.

The complexity $compl(f)$ of $f$ is defined as the complexity of the associated tensor network $\{C^\alpha\}_{\alpha\in T}$.

- Number of parameters (full tensor network)

$$compl_{\mathcal{F}}(f) = \sum_\alpha \text{number\_of\_entries}(C^\alpha)$$

- Number of non-zero parameters (sparse tensor network)

$$compl_{\mathcal{S}}(f) = \sum_\alpha \|C^\alpha\|_0$$

# Complexity measures and corresponding approximation tools

The complexity $compl(f)$ of $f$ is defined as the complexity of the associated tensor network $\{C^\alpha\}_{\alpha \in T}$.

- Number of parameters (full tensor network)

$$compl_{\mathcal{F}}(f) = \sum_{\alpha} \text{number\_of\_entries}(C^\alpha)$$

- Number of non-zero parameters (sparse tensor network)

$$compl_{\mathcal{S}}(f) = \sum_{\alpha} \|C^\alpha\|_0$$

Complexity measures $compl_{\mathcal{F}}$ and $compl_{\mathcal{S}}$ yield two different approximation tools

$$\Phi_n^{\mathcal{F}} \quad \text{and} \quad \Phi_n^{\mathcal{S}}$$

such that

$$\Phi_n^{\mathcal{F}} \subset \Phi_n^{\mathcal{S}} \subset \Phi_{a+bn^2}^{\mathcal{F}}$$

## Approximation theory of tree tensor networks

Given a function $f$ from a Banach space $X$, the best approximation error of $f$ by an element of $\Phi_n$ is

$$E(f, \Phi_n)_X := \inf_{g \in \Phi_n} \|f - g\|_X$$

Fundamental questions are:

- does $E(f, \Phi_n)_X$ converge to 0 for any $f$ ?
  (universality)
- does a best approximation exist ?
  (proximinality)
- how fast does it converge for functions from classical function classes ?
  (expressivity)
- what are the functions for which $E(f, \Phi_n)_X$ converges with some given rate ?
  (characterization of approximation classes)

# Outline

# Universality

First note that for any algebraic feature tensor space $V$, and any tree $T$,

$$\bigcup_r \mathcal{T}_r^T(V) = V.$$

so the question of universality of tree tensor networks boils down to conditions on the tensor feature spaces.

# Universality

First note that for any algebraic feature tensor space $V$, and any tree $T$,

$$\bigcup_r \mathcal{T}_r^T(V) = V.$$

so the question of universality of tree tensor networks boils down to conditions on the tensor feature spaces.

- Consider the first family of approximation tools with variable feature spaces $V_N$, $N \in \mathbb{N}^d$.

  If $\bigcup_N V_N$ is dense in $X$, then the tools are universal for functions in $X$.

  In particular, this is true for $X = L^p((0,1)^d)$, $p < \infty$, and for polynomial or splines spaces $V_N$.

First note that for any algebraic feature tensor space $V$, and any tree $T$,

$$\bigcup_r \mathcal{T}_r^T(V) = V.$$

so the question of universality of tree tensor networks boils down to conditions on the tensor feature spaces.

- Consider the first family of approximation tools with variable feature spaces $V_N$, $N \in \mathbb{N}^d$.

  If $\bigcup_N V_N$ is dense in $X$, then the tools are universal for functions in $X$.

  In particular, this is true for $X = L^p((0,1)^d)$, $p < \infty$, and for polynomial or splines spaces $V_N$.

- Consider the second family of approximation tools using tensorization.

  If $\bigcup_L V_L$ is dense in $X$, then the tools are universal for functions in $X$.

  In particular, this is true for $X = L^p((0,1)^d)$, $p < \infty$, assuming that $S$ contains the function one.

For any tree $T$, any $T$-rank $r$, and any finite dimensional tensor space $V$ of $X$, $\mathcal{T}_r^T(V)$ is a closed set in $V$.

$\Phi_n^{\mathcal{F}}$ (full tensor networks) is a finite union of such sets, all contained in a single finite dimensional space $V^*$. Then $\Phi_n^{\mathcal{F}}$ is a closed set of a finite dimensional space $V^*$ and is therefore proximinal in $X$.

However, $\Phi_n^{\mathcal{F}}$ (sparse tensor networks) is not closed.

Different ways to analyse the expressivity of tree tensor networks

- Exploit known results on other approximation tools and estimate the complexity to encode these tools using tree tensor networks.
- Directly encode a function using tree tensor networks (with controlled errors)
- Analyse the convergence of bilinear approximations

$$u(x_\alpha, x_{\alpha^c}) \approx \sum_{k=1}^{r_\alpha} u_k^\alpha(x_\alpha) u_k^{\alpha^c}(x_{\alpha^c})$$

or the approximability of partial evaluations $u(\cdot, x_{\alpha^c})$ by linear approximation spaces of dimension $r_\alpha$.

# Encoding polynomials and splines

## Polynomials

The tensorization of a polynomial of degree $p$ has all ranks bounded by $p+1$.

## Trigonometric polynomials

The tensorization of the function $\cos(\omega x + \varphi)$ has all ranks equal to 2.

Then the tensorization of a trigonometric polynomial of degree $p$ has all ranks bounded by $2p+1$.

## Free knot splines

A spline $\varphi$ of degree $p$ over $N$ $b$-adic intervals forming a partition of $[0,1)$ is such that

$$\mathrm{rank}_{\{1,\dots,\nu\}}(\varphi) \le \begin{cases} p+N, & 1 \le \nu < \ell. \\ p+1, & \ell \le \nu \le L. \end{cases}$$

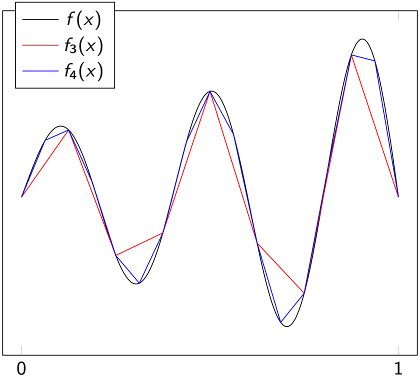
where $b^{-\ell}$ is the minimal length of intervals.

# Encoding polynomials and splines

## Ranks of interpolants

For a function $f$ and its interpolation $f_L$ onto $V_L$, the space of piecewise polynomials of degree $m$ on a uniform partition of $b^L$ intervals, it holds

$$\mathrm{rank}_\alpha(\boldsymbol{f}_L) \leq \mathrm{rank}_\alpha(\boldsymbol{f})$$



# Encoding multi-resolution analysis

For a function $\psi : \mathbb{R} \to \mathbb{R}$ supported on $[0, 1]$, we define its level $\ell$ *b*-adic dilation, shifted by $j = 0, \ldots, b^L - 1$,

$$\psi_{\ell,j}(x) = \psi(b^\ell x - j)$$

Its tensorization at level $\ell$ is an elementary (rank-one) tensor

$$T_{b,\ell}\psi_{\ell,j} = e_{j_1} \otimes \ldots e_{j_\ell} \otimes \psi$$

with $j = [j_1, \ldots, j_\ell]_b$ and $e_k$ the canonical basis vectors in $\mathbb{R}^b$.

# Encoding multi-resolution analysis

For a function $\psi : \mathbb{R} \to \mathbb{R}$ supported on $[0,1]$, we define its level $\ell$ *b*-adic dilation, shifted by $j = 0, \dots, b^L - 1$,

$$\psi_{\ell,j}(x) = \psi(b^\ell x - j)$$

Its tensorization at level $\ell$ is an elementary (rank-one) tensor

$$T_{b,\ell}\psi_{\ell,j} = e_{j_1} \otimes \dots e_{j_\ell} \otimes \psi$$

with $j = [j_1, \dots, j_\ell]_b$ and $e_k$ the canonical basis vectors in $\mathbb{R}^b$.

Its tensorization at level $L \geq l$ is

$$T_{b,L}\psi_{\ell,j} = e_{j_1} \otimes \dots e_{j_\ell} \otimes (T_{b,L-\ell}\psi)$$

The (approximate) encoding of $\psi_{\ell,j}$ boils down to the (approximate) encoding of the mother function $\psi$ with tensor networks.

In particular, if $\psi$ is a (piecewise) polynomial, $\psi_{\ell,j}$ is encoded at precision $\epsilon$ using tensorization at level $L = \ell + O(\log(\epsilon^{-1}))$.

This yields a very efficient encoding of piecewise polynomial MRAs (B-spline wavelets).

# Approximation of functions from Besov spaces $B^\alpha_q(L^p)$

From results on spline approximation and their encoding with tensor networks, we obtain

**Theorem**

*Let $f \in B^\alpha_q(L^p)$ with $\alpha > 0$ and $0 < p, q \leq \infty$. Then*

$$E(f, \Phi_n^{\mathcal{F}})_{L^p} \leq C n^{-\alpha/d} |f|_{B^\alpha_\infty(L^p)}$$

- Tensor networks achieve optimal rates for any Besov regularity order (measured in $L^p$ norm).
- They perform as well as optimal linear approximation tools (e.g. splines), without requiring to adapt the tool to the regularity order $\alpha$.
- The depth (resolution $L$) of the network is crucial to capture extra regularity ($\alpha > m + 1$).

# Approximation of functions from Besov spaces $B_q^\alpha(L^\tau)$

Now consider the harder problem of approximating functions from Besov spaces $B_q^\alpha(L^\tau)$ where regularity is measured in a $L^\tau$-norm weaker than $L^p$-norm.

From results on best $n$-term approximation using dilated splines, we obtain

**Theorem**

*Let $f \in B_q^\alpha(L^\tau)$ with $\alpha > 0$, $0 < q \leq \tau < p < \infty$, $1 \leq p < \infty$ and*

$$\frac{\alpha}{d} > \frac{1}{\tau} - \frac{1}{p}.$$

*Then*

$$E(f, \Phi_n^{\mathcal{S}})_{L^p} \leq C n^{-\tilde{\alpha}/d} |f|_{B_q^\alpha(L^\tau)}, \quad E(f, \Phi_n^{\mathcal{F}})_{L^p} \leq C n^{-\tilde{\alpha}/(2d)} |f|_{B_q^\alpha(L^\tau)},$$

*for arbitrary $\tilde{\alpha} < \alpha$.*

## Approximation of functions from Besov spaces $B^\alpha_q(L^\tau)$

Now consider the harder problem of approximating functions from Besov spaces $B^\alpha_q(L^\tau)$ where regularity is measured in a $L^\tau$-norm weaker than $L^p$-norm.

From results on best $n$-term approximation using dilated splines, we obtain

### Theorem

*Let $f \in B^\alpha_q(L^\tau)$ with $\alpha > 0$, $0 < q \leq \tau < p < \infty$, $1 \leq p < \infty$ and*

$$\frac{\alpha}{d} > \frac{1}{\tau} - \frac{1}{p}.$$

*Then*

$$E(f, \Phi^{\mathcal{S}}_n)_{L^p} \leq C n^{-\tilde{\alpha}/d} |f|_{B^\alpha_q(L^\tau)}, \quad E(f, \Phi^{\mathcal{F}}_n)_{L^p} \leq C n^{-\tilde{\alpha}/(2d)} |f|_{B^\alpha_q(L^\tau)},$$

*for arbitrary $\tilde{\alpha} < \alpha$.*

- Sparse tensor networks achieve arbitrarily close to optimal rates in $O(n^{-\alpha/d})$ for functions with any Besov smoothness $\alpha$ (measured in $L^\tau$ norm), without the need to adapt the tool to the regularity order $\alpha$.
- Here depth and sparsity are crucial for obtaining near to optimal performance.
- Full tensor networks have slightly lower performance in $O(n^{-\alpha/(2d)})$.

# High-dimensional approximation

- For Besov spaces $B^{\alpha}_{q}(L^{p})$, tensor networks achieve (near to) optimal rate in $O(n^{-\alpha/d})$ which deteriorates with $d$, that is the curse of dimensionality.

# High-dimensional approximation

- For Besov spaces $B^{\alpha}_{q}(L^{p})$, tensor networks achieve (near to) optimal rate in $O(n^{-\alpha/d})$ which deteriorates with $d$, that is the curse of dimensionality.
- For Besov spaces with mixed smoothness $MB^{\alpha}_{q}(L^{p})$ , sparse tensor networks achieve near to optimal performance in $O(n^{-\alpha}\log(n)^{d})$. But still the curse of dimensionality.

- For Besov spaces $B_q^\alpha(L^p)$, tensor networks achieve (near to) optimal rate in $O(n^{-\alpha/d})$ which deteriorates with $d$, that is the curse of dimensionality.
- For Besov spaces with mixed smoothness $MB_q^\alpha(L^p)$ , sparse tensor networks achieve near to optimal performance in $O(n^{-\alpha}\log(n)^d)$. But still the curse of dimensionality.
- For Besov spaces with anisotropic smoothness $AB_q^{\boldsymbol{\alpha}}(L^p)$, sparse tensor networks also achieve near to optimal rates in $O(n^{-s(\boldsymbol{\alpha})/d})$ with

$$s(\boldsymbol{\alpha})/d = (\alpha_1^{-1} + \ldots + \alpha_d^{-1})^{-1}$$

  the aggregated smoothness. Curse of dimensionality can be circumvented with sufficient anisotropy.

# High-dimensional approximation

- For Besov spaces $B^{\alpha}_{q}(L^{p})$, tensor networks achieve (near to) optimal rate in $O(n^{-\alpha/d})$ which deteriorates with $d$, that is the curse of dimensionality.
- For Besov spaces with mixed smoothness $MB^{\alpha}_{q}(L^{p})$ , sparse tensor networks achieve near to optimal performance in $O(n^{-\alpha}\log(n)^{d})$. But still the curse of dimensionality.
- For Besov spaces with anisotropic smoothness $AB^{\boldsymbol{\alpha}}_{q}(L^{p})$, sparse tensor networks also achieve near to optimal rates in $O(n^{-s(\boldsymbol{\alpha})/d})$ with

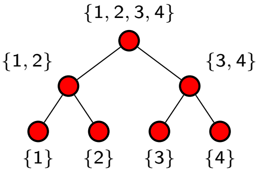
  $$s(\boldsymbol{\alpha})/d = (\alpha_1^{-1} + \ldots + \alpha_d^{-1})^{-1}$$

  the aggregated smoothness. Curse of dimensionality can be circumvented with sufficient anisotropy.
- Curse of dimensionality can be circumvented for non usual function classes such as compositions of smooth functions

# Compositional functions

Consider a tree-structured composition of smooth functions $\{f_\alpha : \alpha \in T\}$, see [Mhaskar, Liao, Poggio 2016] for deep neural networks, and [Bachmayr Nouy and Schneider 2021] for tree tensor networks.

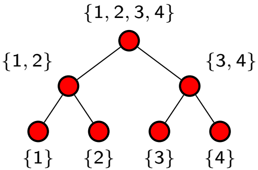
$$f_{1,2,3,4}\left(f_{1,2}\left(f_1(x_1), f_2(x_2)\right), f_{3,4}\left(f_3(x_3), f_4(x_4)\right)\right)$$



# Compositional functions

Consider a tree-structured composition of smooth functions $\{f_\alpha : \alpha \in T\}$, see [Mhaskar, Liao, Poggio 2016] for deep neural networks, and [Bachmayr Nouy and Schneider 2021] for tree tensor 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 $\|f_\alpha\|_{L^\infty} \leq 1$ and $\|f_\alpha\|_{W^{1,\infty}} \leq B$, the complexity to achieve an accuracy $\epsilon$

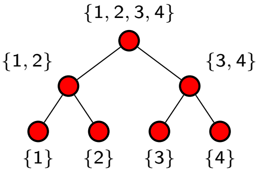
$$n(\epsilon) \lesssim \epsilon^{-3/k}(L+1)^3 B^{3L} d^{1+3/2k}$$

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

# Compositional functions

Consider a tree-structured composition of smooth functions $\{f_\alpha : \alpha \in T\}$, see [Mhaskar, Liao, Poggio 2016] for deep neural networks, and [Bachmayr Nouy and Schneider 2021] for tree tensor 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 $\|f_\alpha\|_{W^{1,\infty}} \le B$, the complexity to achieve an accuracy $\epsilon$

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

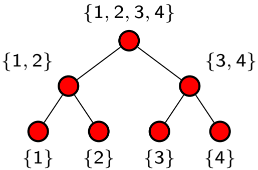
with $L = \log_2(d)$ for a balanced tree and $L + 1 = d$ for a linear tree.

- Bad influence of the depth through the norm $B$ of functions $f_\alpha$ (roughness).

## Compositional functions

Consider a tree-structured composition of smooth functions $\{f_\alpha : \alpha \in T\}$, see [Mhaskar, Liao, Poggio 2016] for deep neural networks, and [Bachmayr Nouy and Schneider 2021] for tree tensor 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 $\|f_\alpha\|_{L^\infty} \leq 1$ and $\|f_\alpha\|_{W^{1,\infty}} \leq B$, the complexity to achieve an accuracy $\epsilon$

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

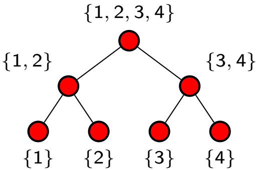
with $L = \log_2(d)$ for a balanced tree and $L + 1 = d$ for a linear tree.

- Bad influence of the depth through the norm $B$ of functions $f_\alpha$ (roughness).
- For a balanced tree, complexity scales polynomially in $d$: no curse of dimensionality !

# Compositional functions

Consider a tree-structured composition of smooth functions $\{f_\alpha : \alpha \in T\}$, see [Mhaskar, Liao, Poggio 2016] for deep neural networks, and [Bachmayr Nouy and Schneider 2021] for tree tensor 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 $\|f_\alpha\|_{L^\infty} \leq 1$ and $\|f_\alpha\|_{W^{1,\infty}} \leq B$, the complexity to achieve an accuracy $\epsilon$

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

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 $f_\alpha$ (roughness).
- For a balanced tree, complexity scales polynomially in $d$: no curse of dimensionality !
- For $B \leq 1$, the complexity only scales polynomially in $d$ whatever the tree: no curse of dimensionality !

# More regularity, analytic functions

For function $f : [0,1]$ with analytic extension on an open complex domain

$$D_\rho = \{z \in \mathbb{C} : dist(z, [0,1]) < \frac{\rho - 1}{2}\}, \quad \rho > 1,$$

we obtain an exponential convergence

$$E(f, \Phi_n^{\mathcal{F}})_{L^\infty} \leq C\gamma^{-n^{\mathbf{1/3}}},$$

with $\gamma = \min\{\rho, b^{(m+1)/b}\}$.

For function $f : [0,1]$ with analytic extension on an open complex domain

$$D_\rho = \{z \in \mathbb{C} : dist(z,[0,1]) < \frac{\rho-1}{2}\}, \quad \rho > 1,$$

we obtain an exponential convergence

$$E(f, \Phi_n^{\mathcal{F}})_{L^\infty} \le C\gamma^{-n^{1/3}},$$

with $\gamma = \min\{\rho, b^{(m+1)/b}\}$.

The proof relies on the approximation of analytic functions with polynomials and the encoding of polynomials with tree tensor networks: a chebychev polynomial $p$ of deree $\bar{m}$ is such that

$$\|f-p\|_{L^\infty} \le \frac{2}{\rho-1}\|f\|_{L^\infty(D_\rho)}\rho^{-\bar{m}}$$

A polynomial of degree $\bar{m}$ can be approximated by $\varphi$ in $\Phi_{L,r,m}$ with an error in $O(b^{-L(m+1)})$, so that

$$\|f-\varphi\|_{L^\infty} \lesssim \rho^{-\bar{m}} + b^{-L(m+1)}$$

We obtain the result by choosing $\bar{m} \sim n^{1/3}$ and $L \sim b^{-1}n^{1/3}$, so that $compl_{\mathcal{F}}(\varphi) \le n$.

# Analytic functions with singularities

Consider the approximation of $u(x) = x^\alpha$, $0 < \alpha \leq 1$, in $L^\infty$.

- Piecewise constant linear approximation.

$$u \in B^\alpha_\infty(L^\infty), \quad u \notin B^\beta_\infty(L^\infty) \quad \text{for } \beta > \alpha,$$

  and a piecewise constant approximation on a uniform mesh with $n$ elements gives a convergence in $O(n^{-\alpha})$ in $L^\infty$,

# Analytic functions with singularities

Consider the approximation of $u(x) = x^\alpha$, $0 < \alpha \leq 1$, in $L^\infty$.

- Piecewise constant linear approximation.

$$u \in B^\alpha_\infty(L^\infty), \quad u \notin B^\beta_\infty(L^\infty) \quad \text{for } \beta > \alpha,$$

  and a piecewise constant approximation on a uniform mesh with $n$ elements gives a convergence in $O(n^{-\alpha})$ in $L^\infty$,

- Piecewise constant nonlinear approximation.

$$u \in BV \subset B^1_\infty(L^1),$$

  and a piecewise constant approximation on an optimal mesh with $n$ elements gives a convergence in $O(n^{-1})$ in $L^\infty$,

# Analytic functions with singularities

Consider the approximation of $u(x) = x^\alpha$, $0 < \alpha \leq 1$, in $L^\infty$.

- Piecewise constant linear approximation.

$$u \in B^\alpha_\infty(L^\infty), \quad u \notin B^\beta_\infty(L^\infty) \quad \text{for } \beta > \alpha,$$

  and a piecewise constant approximation on a uniform mesh with $n$ elements gives a convergence in $O(n^{-\alpha})$ in $L^\infty$,

- Piecewise constant nonlinear approximation.

$$u \in BV \subset B^1_\infty(L^1),$$

  and a piecewise constant approximation on an optimal mesh with $n$ elements gives a convergence in $O(n^{-1})$ in $L^\infty$,

- Piecewise constant approximation and tensor networks.
  A piecewise constant approximation on a uniform mesh with $2^L$ elements exploiting low-rank structures gives an exponential convergence

$$E(f, \Phi_n^{\mathcal{F}}) \leq C\beta^{-n^\gamma}$$

  Achieves almost the performance of *h-p* methods [Kazeev and Schwab].

# Beyond smoothness

Consider the Weierstrass function, continuous but nowhere differentiable

$$f(x) = \sum_{k=0}^{\infty} a^{-\alpha k} \cos(a^k \pi x), \quad a > 0, \quad 0 < \alpha \leq 1,$$

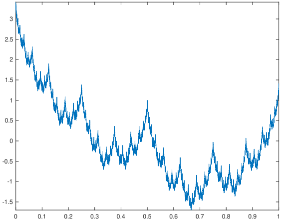



Figure: Weierstrass function for $\alpha = 1/2$ , $a = 2$

We have an exponential convergence in $L^\infty$-norm

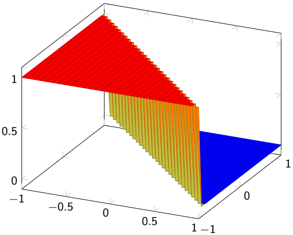
$$E(f, \Phi_n^{\mathcal{F}})_{L^\infty} \lesssim \beta^{-n^{1/3}}$$

An error $\epsilon$ is achieved with resolution $L \sim \log(\epsilon^{-1})$, ranks $\sim \log(\epsilon^{-1})$ and complexity $n \sim \log(\epsilon^{-1})^3$

# Discontinuous functions: the power of tensorization

Consider the problem of approximating the bivariate function on $(-1,1)^2$

$$u(x,t) = \begin{cases} 1 & \text{if } x+t<0 \\ 0 & \text{if } x+t \geq 0 \end{cases}$$




The manifold $K = \{u(\cdot,t) : t \in (-1,1)\}$ contains the indicator functions $1_{[-1,x_i]}(x)$, $x_i = -1 + 2i/m$. Therefore the balanced convex hull of $K$ contains the orthogonal system $S = \{\psi_i(x) = \frac{1}{2} 1_{(x_i,x_{i+1}]}(x) : 1 \leq i \leq m\}$ with $\|\psi_i\|_{L^2} = (2m)^{-1/2}$ and by taking $m = 2n$, we deduce

$$d_n(K)_{L^2} \geq 1/(2\sqrt{2})n^{-1/2},$$

so that the best rank-$n$ approximation

$$u_n(x,t) = \sum_{i=1}^{n} v_i(x) w_i(t)$$

does not converge better than $\|u - u_n\|_{L^2} \gtrsim n^{-1/2}$.

# Discontinuous functions: the power of tensorization

A piecewise constant interpolant $u^L$ on a uniform grid with mesh size $2^{-L}$ is such that

$$\|u - u^L\|_{L^2} \leq meas(\{(x,t) : u \neq u^L\})^{1/2} \leq 2^{1/2} 2^{-L/2}$$

Using a tensorization $\tilde{\boldsymbol{u}}^L(i_1^x, ..., i_L^x, i_1^t, ..., i_L^t)$, we have

$$rank_{\{1,\ldots,L\}}(\tilde{\boldsymbol{u}}^L) = \operatorname{rank} u_L \sim 2^L$$

that means an encoding complexity in tensor train format $compl(\tilde{\boldsymbol{u}}^L) \gtrsim 2^{2L}$, which yields an approximation error $\gtrsim n^{-1/4}$.

However, the tensorization $\boldsymbol{u}^L(i_1^x, i_1^t, ..., i_L^x, i_L^t)$ of $u^L(x,t)$ satisfies

$$rank_{\{1,\ldots,\nu\}}(\boldsymbol{u}^L) \leq 3$$

for all $\nu$. Therefore, using tensor train format, $compl(u^L) \leq 36L$ and

$$E(u, \Phi_n^{\mathcal{F}})_{L^2} \leq 2^{1/2} 2^{-n/72}.$$

# Outline

# Canonical versus tree-based format

Consider a finite dimensional tensor space $V = V^1 \otimes \ldots \otimes V^d$ with $\dim(V_\nu) = \mathbb{R}^N$, which is identified with $\mathbb{R}^{N\times\cdots\times N}$. Denote by $\mathcal{R}_r = \{v : \text{rank}(v) \leq r\}$ and $\mathcal{T}_r^T = \{v : \text{rank}_\alpha(v) \leq r, \alpha \in T\}$.

- From canonical format to tree-based format.
  For any $v$ in $V$ and any $\alpha \subset D$, the $\alpha$-rank is bounded by the canonical rank:

$$\text{rank}_\alpha(v) \leq \text{rank}(v).$$

  Therefore, for any tree $T$,

$$\mathcal{R}_r \subset \mathcal{T}_r^T,$$

  so that an element in $\mathcal{R}_r$ with storage complexity $O(dNr)$ admits a representation in $\mathcal{T}_r^T$ with a storage complexity $O(dNr + dr^{s+1})$ where $s$ is the arity of the tree $T$.

# Canonical versus tree-based format

Consider a finite dimensional tensor space $V = V^1 \otimes \ldots \otimes V^d$ with $\dim(V_\nu) = \mathbb{R}^N$, which is identified with $\mathbb{R}^{N\times\cdots\times N}$. Denote by $\mathcal{R}_r = \{v : \mathrm{rank}(v) \leq r\}$ and $\mathcal{T}_r^T = \{v : \mathrm{rank}_\alpha(v) \leq r, \alpha \in T\}$.

- From canonical format to tree-based format.
  For any $v$ in $V$ and any $\alpha \subset D$, the $\alpha$-rank is bounded by the canonical rank:
  $$\mathrm{rank}_\alpha(v) \leq \mathrm{rank}(v).$$
  Therefore, for any tree $T$,
  $$\mathcal{R}_r \subset \mathcal{T}_r^T,$$
  so that an element in $\mathcal{R}_r$ with storage complexity $O(dNr)$ admits a representation in $\mathcal{T}_r^T$ with a storage complexity $O(dNr + dr^{s+1})$ where $s$ is the arity of the tree $T$.

- From tree-based format to canonical format. For a balanced or linear binary tree, the subset
  $$S = \{v \in \mathcal{T}_r^T : \mathrm{rank}(v) < q^{d/2}\}, \quad q = \min\{N, r\},$$
  is of Lebesgue measure 0.

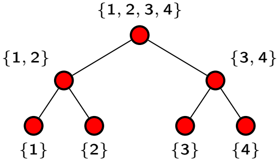
  Then a typical element $v \in \mathcal{T}_r^T$ with storage complexity of order $dNr + dr^3$ admits a representation in canonical format with a storage complexity of order $dNq^{d/2}$.

## Influence of the tree

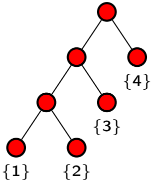
- For some functions, the choice of tree is not crucial. For example, an additive function
$$u_1(x_1) + \ldots + u_d(x_d)$$
has $\alpha$-ranks equal to 2 whatever $\alpha \subset D$.

# Influence of the tree

- For some functions, the choice of tree is not crucial. For example, an additive function
$$u_1(x_1) + \ldots + u_d(x_d)$$
has $\alpha$-ranks equal to 2 whatever $\alpha \subset D$.
- But usually, different trees lead to different complexities of representations.




$T^B$ (Balanced tree)




$T^L$ (Linear tree)

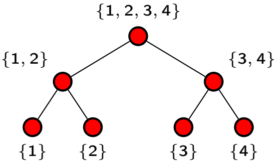
- If $\text{rank}_{T^L}(u) \leq r$ then $\text{rank}_{T^B}(u) \leq r^2$
- If $\text{rank}_{T^B}(u) \leq r$ then $\text{rank}_{T^L}(u) \leq r^{\log_2(d)/2}$

# Influence of the tree

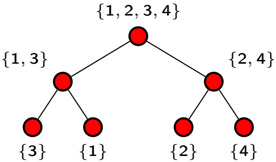
Given a tree $T$ and a permutation $\sigma$ of $D = \{1, \dots, d\}$, we define a tree $T_\sigma$

$$T_\sigma = \{\sigma(\alpha) : \alpha \in T\}$$

having the same structure as $T$ but different nodes.




$T$




$T_\sigma$ with $\sigma = (3, 1, 2, 4)$

If $\text{rank}_T(u) \leq r$ then $\text{rank}_{T_\sigma}(u)$ typically depends on $d$.

- Consider the Henon-Heiles potential

$$u(x) = \frac{1}{2}\sum_{i=1}^{d} x_i^2 + 0.2\sum_{i=1}^{d-1}(x_i x_{i+1}^2 - x_i^3) + \frac{0.2^2}{16}\sum_{i=1}^{d-1}(x_i^2 + x_{i+1}^2)^2$$

Using a linear tree $T = \{\{1\},\{2\},\ldots,\{d\},\{1,2\},\{1,2,3\},\ldots,\{1,\ldots,d-1\},D\}$,

$$\text{rank}_T(u) \leq 4, \quad \textit{storage}(u) = O(d)$$

but for the permutation

$$\sigma = (1,3,\ldots,d-1,2,4,\ldots,d) \qquad (\star)$$

and the corresponding linear tree $T_\sigma$,

$$\text{rank}_{T_\sigma}(u) \leq 2d+1, \quad \textit{storage}(u) = O(d^3).$$

## Influence of the tree

- Consider the Henon-Heiles potential

$$u(x) = \frac{1}{2}\sum_{i=1}^{d} x_i^2 + 0.2\sum_{i=1}^{d-1}(x_i x_{i+1}^2 - x_i^3) + \frac{0.2^2}{16}\sum_{i=1}^{d-1}(x_i^2 + x_{i+1}^2)^2$$

Using a linear tree $T = \{\{1\},\{2\},\ldots,\{d\},\{1,2\},\{1,2,3\},\ldots,\{1,\ldots,d-1\},D\}$,

$$\mathrm{rank}_T(u) \leq 4, \quad \mathit{storage}(u) = O(d)$$

but for the permutation

$$\sigma = (1,3,\ldots,d-1,2,4,\ldots,d) \tag{$\star$}$$

and the corresponding linear tree $T_\sigma$,

$$\mathrm{rank}_{T_\sigma}(u) \leq 2d+1, \quad \mathit{storage}(u) = O(d^3).$$

- For a typical tensor in $\mathcal{T}_r^T$ with $T$ a binary tree, its representation in tree based format with tree $T_\sigma$, with $\sigma$ as in $(\star)$, has a complexity scaling exponentially with $d$.

## Influence of the tree

- Consider the Henon-Heiles potential

$$u(x) = \frac{1}{2}\sum_{i=1}^{d} x_i^2 + 0.2\sum_{i=1}^{d-1}(x_i x_{i+1}^2 - x_i^3) + \frac{0.2^2}{16}\sum_{i=1}^{d-1}(x_i^2 + x_{i+1}^2)^2$$

Using a linear tree $T = \{\{1\},\{2\},\ldots,\{d\},\{1,2\},\{1,2,3\},\ldots,\{1,\ldots,d-1\},D\}$,

$$\mathrm{rank}_T(u) \leq 4, \quad storage(u) = O(d)$$

but for the permutation

$$\sigma = (1,3,\ldots,d-1,2,4,\ldots,d) \qquad (\star)$$

and the corresponding linear tree $T_\sigma$,

$$\mathrm{rank}_{T_\sigma}(u) \leq 2d+1, \quad storage(u) = O(d^3).$$

- For a typical tensor in $\mathcal{T}_r^T$ with $T$ a binary tree, its representation in tree based format with tree $T_\sigma$, with $\sigma$ as in $(\star)$, has a complexity scaling exponentially with $d$.
- As an example, consider the function $u(x,t) = 1_{x+t<0}$identified (through tensorization) with tensors $\boldsymbol{u}(i_1^x,\ldots,i_L^x,y^x,i_1^t,\ldots,i_L^t,y^t)$ and $\boldsymbol{u}(i_1^x,i_1^y,\ldots,i_L^x,i_L^y,y^x,y^t)$. Huge impact of the ordering !

- 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 $r$.

# Influence of the tree

- 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 $r$.

  - 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 $r^4$.

# Influence of the tree

- 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 $r$.

  - 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 $d$.

# Influence of the tree

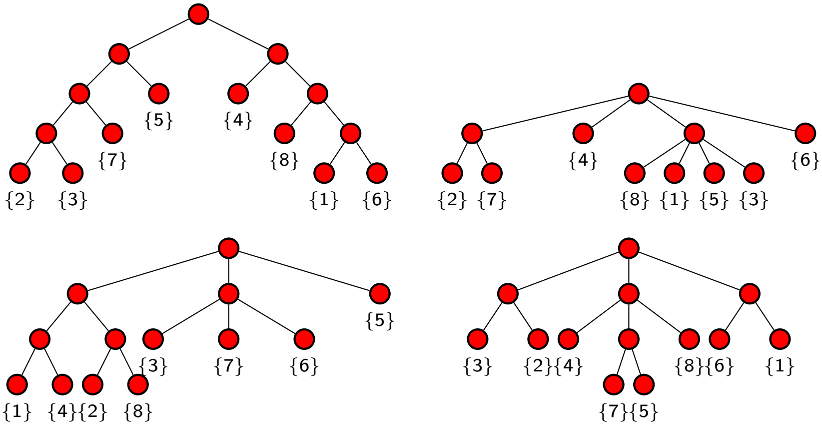
- Consider the probability distribution $f(x) = \mathbb{P}(X = x)$ of a Markov chain $X = (X_1, \dots, X_d)$ given by

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

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

  - 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 $T_\sigma$ 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 $d$.

A combinatorial problem...



# Outline

# Properties of tree tensor networks

We here consider approximation tools $(\Phi_n)_{n\geq 1}$ based on tensorization and tensor train format (with or without sparsity).
They satisfy

(P1) $\Phi_0 = \{0\}$, $0 \in \Phi_n$

(P2) $a\Phi_n = \Phi_n$ for any $a \in \mathbb{R} \setminus \{0\}$ (cone)

(P3) $\Phi_n \subset \Phi_{n+1}$ (nestedness)

(P4) $\Phi_n + \Phi_n \subset \Phi_{cn}$ for some constant $c$ (not too nonlinear)

# Properties of tree tensor networks

We here consider approximation tools $(\Phi_n)_{n\geq 1}$ based on tensorization and tensor train format (with or without sparsity).
They satisfy

(P1) $\Phi_0 = \{0\}$, $0 \in \Phi_n$

(P2) $a\Phi_n = \Phi_n$ for any $a \in \mathbb{R} \setminus \{0\}$ (cone)

(P3) $\Phi_n \subset \Phi_{n+1}$ (nestedness)

(P4) $\Phi_n + \Phi_n \subset \Phi_{cn}$ for some constant $c$ (not too nonlinear)

For $X = L^p$, they further satisfy

(P5) $\bigcup_n \Phi_n$ is dense in $L^p$ for $0 < p < \infty$ (universality),

(P6) for each $f \in L^p$ for $0 < p \leq \infty$, there exists a best approximation in $\Phi_n^{\mathcal{F}}$ (proximinal sets). However, $\Phi_n^{\mathcal{S}}$ is not closed.

# Approximation classes

For an approximation tool $\Phi = (\Phi_n)_{n\in\mathbb{N}}$, we define for any $\alpha > 0$ the approximation class

$$A^\alpha_\infty(L^p) := A^\alpha_\infty(L^p, \Phi)$$

of functions $f \in L^p$ such that

$$E(f, \Phi_n)_{L^p} \leq Cn^{-\alpha}$$

# Approximation classes

For an approximation tool $\Phi = (\Phi_n)_{n\in\mathbb{N}}$, we define for any $\alpha > 0$ the approximation class

$$A^{\alpha}_{\infty}(L^p) := A^{\alpha}_{\infty}(L^p, \Phi)$$

of functions $f \in L^p$ such that

$$E(f, \Phi_n)_{L^p} \leq Cn^{-\alpha}$$

- Properties (P1)-(P4) of $\Phi$ imply that $A^{\alpha}_{\infty}(L^p)$ is a quasi-Banach space with quasi-semi-norm

$$|f|_{A^{\alpha}_{\infty}} := \sup_{n\geq 1} n^{\alpha} E(f, \Phi_n)_{L^p}$$

# Approximation classes

For an approximation tool $\Phi = (\Phi_n)_{n\in\mathbb{N}}$, we define for any $\alpha > 0$ the approximation class

$$A^\alpha_\infty(L^p) := A^\alpha_\infty(L^p, \Phi)$$

of functions $f \in L^p$ such that

$$E(f, \Phi_n)_{L^p} \leq Cn^{-\alpha}$$

- Properties (P1)-(P4) of $\Phi$ imply that $A^\alpha_\infty(L^p)$ is a quasi-Banach space with quasi-semi-norm

$$|f|_{A^\alpha_\infty} := \sup_{n\geq 1} n^\alpha E(f, \Phi_n)_{L^p}$$

- Full and sparse complexity measures yield two different approximation spaces

$$\mathcal{F}^\alpha_\infty(L^p) = A^\alpha_\infty(L^p, \Phi^{\mathcal{F}}), \quad \mathcal{S}^\alpha_\infty(L^p) = A^\alpha_\infty(L^p, \Phi^{\mathcal{S}})$$

  such that

$$\mathcal{F}^\alpha_\infty(L^p) \hookrightarrow \mathcal{S}^\alpha_\infty(L^p) \hookrightarrow \mathcal{F}^{\alpha/2}_\infty(L^p)$$

# Direct embeddings

From results on the approximation properties for Besov spaces, we have the following results.

- (Linear approximation) For $\alpha > 0$ and $0 < p \leq \infty$,

$$B_q^\alpha(L^p) \hookrightarrow \mathcal{F}_\infty^{\alpha/d}(L^p),$$

$$MB_q^\alpha(L^p) \hookrightarrow \mathcal{S}_\infty^{\alpha}(L^p),$$

$$AB_q^{\boldsymbol{\alpha}}(L^p) \hookrightarrow \mathcal{S}_\infty^{s/d}(L^p)$$

  with $s(\boldsymbol{\alpha}) := d(\alpha_1^{-1} + \ldots + \alpha_d^{-1})^{-1}$.

## Direct embeddings

From results on the approximation properties for Besov spaces, we have the following results.

- (Linear approximation) For $\alpha > 0$ and $0 < p \leq \infty$,

$$B_q^\alpha(L^p) \hookrightarrow \mathcal{F}_\infty^{\alpha/d}(L^p),$$

$$MB_q^\alpha(L^p) \hookrightarrow \mathcal{S}_\infty^{\alpha}(L^p),$$

$$AB_q^{\boldsymbol{\alpha}}(L^p) \hookrightarrow \mathcal{S}_\infty^{s/d}(L^p)$$

  with $s(\boldsymbol{\alpha}) := d(\alpha_1^{-1} + \ldots + \alpha_d^{-1})^{-1}$.

- (Nonlinear approximation) For $\alpha > 0$, $1 \leq p < \infty$, $0 < q \leq \tau < p < \infty$ and $\frac{\alpha}{d} > \frac{1}{\tau} - \frac{1}{p}$,

$$B_q^\alpha(L^\tau) \hookrightarrow \mathcal{S}_\infty^{\tilde{\alpha}/d}(L^p) \hookrightarrow \mathcal{F}_\infty^{\tilde{\alpha}/(2d)}(L^p),$$

$$MB_q^\alpha(L^\tau) \hookrightarrow \mathcal{S}_\infty^{\tilde{\alpha}}(L^p) \hookrightarrow \mathcal{F}_\infty^{\tilde{\alpha}/2}(L^p)$$

  for arbitrary $\tilde{\alpha} < \alpha$, and

$$AB_q^{\boldsymbol{\alpha}}(L^\tau) \hookrightarrow \mathcal{S}_\infty^{\tilde{\alpha}/d}(L^p) \hookrightarrow \mathcal{F}_\infty^{\tilde{\alpha}/(2d)}(L^p)$$

  for arbitrary $\tilde{\alpha} < s(\boldsymbol{\alpha})$.

# Interpolation family

The properties of $\Phi_n$ allow to apply classical results from approximation theory, in particular to deduce from embedding results on $A^{\alpha}_{\infty}(L^p)$ embedding results on interpolation spaces

$$A^{\beta}_{q}(L^p) = (L^p, A^{\alpha}_{\infty}(L^p))_{\beta/\alpha,q}, \quad 0 < \beta < \alpha, \qquad 0 < q \leq \infty$$

that are quasi-Banach spaces with quasi-norm

$$\|f\|_{A^{\alpha}_{q}} = \|f\|_{L^p} + |f|_{A^{\alpha}_{q}}, \quad |f|_{A^{\alpha}_{q}} = \left(\sum_{n=1}^{\infty} n^{-1} \left(n^{\alpha} E(f, \Phi_n)_X\right)^q\right)^{1/q}$$

(functions with faster convergence than those of $A^{\alpha}_{\infty}(L^p)$).

# No inverse embedding

For any $\alpha > 0$, $q \leq \infty$, and any $\beta$,

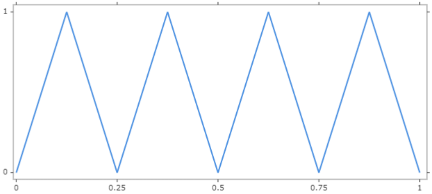
$$\mathcal{F}_q^\alpha(L^p) \not\hookrightarrow B_q^\beta(L^p).$$

That means that approximation classes contain functions that have no smoothness in a classical sense.

Tree tensor networks may be useful for the approximation of functions beyond standard smoothness classes.

# No inverse embedding

This is proved by contradiction by considering the sawtooth function $\varphi_L$ with $2^L$ teeth such that $\varphi_L \in \Phi_n$ with $n \sim L$.




From properties (P1)-(P6), $\mathcal{F}_q^\alpha(L^p)$ satisfies the Berstein inequality, that is

$$\|\varphi\|_{\mathcal{F}_q^\alpha(L^p)} \lesssim n^\alpha \|\varphi\|_{L^p} \quad \forall \varphi \in \Phi_n.$$

Moreover, $\|\varphi_L\|_{L^p} \sim 1$ and $\|\varphi_L\|_{B_q^\beta(L^p)} \gtrsim 2^{\beta L}$. If the embedding were true, we would have

$$2^{\beta n} \lesssim \|\varphi_L\|_{B_q^\beta(L^p)} \lesssim \|\varphi_L\|_{\mathcal{F}_q^\alpha(L^p)} \lesssim n^\alpha,$$

a contradiction.

# The role of depth

Consider the approximation with restricted resolution

$$\Phi_n^{\mathcal{L}} = \{f \in \Phi_n : L(f) \leq \mathcal{L}(n)\}$$

where $L(f)$ is the minimal resolution $L$ such that $f \in V_L$, and $\mathcal{L}$ some growth function.

Since $L(f) \leq n$ for $f \in \Phi_n$, $\Phi_n^{\mathcal{L}} = \Phi_n$ for $\mathcal{L} = n$.

In dimension $d = 1$, for $\mathcal{L}(n) = r \log_b(n) + c$, the following Bernstein inequality holds

$$|f|_{B_\tau^{m+1}(L^\tau)} \lesssim \|f\|_{L^p} b^{c(m+1)} n^{r(m+1)}$$

with $\tau$ the Sobolev embedding number, and $m$ the local polynomial degree. This implies the inverse embedding of the corresponding approximation class

$$A_\infty^\alpha(L^p; (\Phi_n^{\mathcal{L}})) \hookrightarrow B_\tau^{\alpha/(m+1)}(L^\tau)$$

Hence the importance of depth $L$ for going beyond standard regularity classes.

# Overview of results on approximation theory of tensor networks

- Using tensorization of functions, efficient encoding of many approximation tools: polynomials, trigonometric polynomials, free knot splines, multi-resolution analysis (wavelets).

# Overview of results on approximation theory of tensor networks

- Using tensorization of functions, efficient encoding of many approximation tools: polynomials, trigonometric polynomials, free knot splines, multi-resolution analysis (wavelets).
- Using tensorization, no need to adapt the approximation tool to the regularity of functions. For that, allowing deep networks (high resolution) is crucial.

# Overview of results on approximation theory of tensor networks

- Using tensorization of functions, efficient encoding of many approximation tools: polynomials, trigonometric polynomials, free knot splines, multi-resolution analysis (wavelets).
- Using tensorization, no need to adapt the approximation tool to the regularity of functions. For that, allowing deep networks (high resolution) is crucial.
- For analytic functions (with possible singularities), exponential convergence is achieved.

# Overview of results on approximation theory of tensor networks

- Using tensorization of functions, efficient encoding of many approximation tools: polynomials, trigonometric polynomials, free knot splines, multi-resolution analysis (wavelets).
- Using tensorization, no need to adapt the approximation tool to the regularity of functions. For that, allowing deep networks (high resolution) is crucial.
- For analytic functions (with possible singularities), exponential convergence is achieved.
- For Sobolev spaces $W^{\alpha,p}$ or Besov spaces $B^{\alpha}_{q}(L^p)$, tensor networks (full or sparse) achieve optimal rate in $O(n^{-\alpha/d})$.

# Overview of results on approximation theory of tensor networks

- Using tensorization of functions, efficient encoding of many approximation tools: polynomials, trigonometric polynomials, free knot splines, multi-resolution analysis (wavelets).
- Using tensorization, no need to adapt the approximation tool to the regularity of functions. For that, allowing deep networks (high resolution) is crucial.
- For analytic functions (with possible singularities), exponential convergence is achieved.
- For Sobolev spaces $W^{\alpha,p}$ or Besov spaces $B^{\alpha}_{q}(L^{p})$, tensor networks (full or sparse) achieve optimal rate in $O(n^{-\alpha/d})$.
- For Besov spaces $B^{\alpha}_{q}(L^{\tau})$ $(\tau < p)$, sparse tensor networks achieve arbitrary close to optimal rate in $O(n^{-\alpha/d})$, while full tensor networks achieve a rate arbitrarily close to $O(n^{-\alpha/(2d)})$.

- For Besov spaces with mixed dominating smoothness $MB^{\alpha}_{q}(L^p)$ , sparse tensor networks achieve near to optimal performance in $O(n^{-\alpha}\log(n)^d)$.

- For Besov spaces with mixed dominating smoothness $MB_q^{\alpha}(L^p)$ , sparse tensor networks achieve near to optimal performance in $O(n^{-\alpha}\log(n)^d)$.
- For Besov spaces with anisotropic smoothness $AB_q^{\boldsymbol{\alpha}}(L^p)$, sparse tensor networks also achieve near to optimal rates in $O(n^{-s(\boldsymbol{\alpha})/d})$ with

$$s(\boldsymbol{\alpha})/d = (\alpha_1^{-1} + \ldots + \alpha_d^{-1})^{-1}$$

  the aggregated smoothness. Curse of dimensionality can be circumvented with sufficient anisotropy.

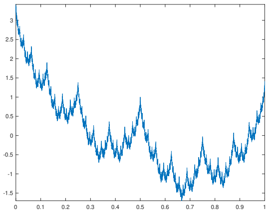
# Overview of results on approximation theory of tensor networks

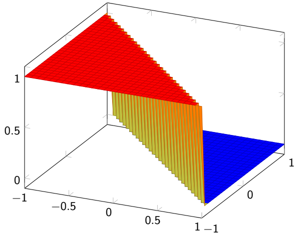
- For Besov spaces with mixed dominating smoothness $MB^{\alpha}_q(L^p)$ , sparse tensor networks achieve near to optimal performance in $O(n^{-\alpha}\log(n)^d)$.
- For Besov spaces with anisotropic smoothness $AB^{\boldsymbol{\alpha}}_q(L^p)$, sparse tensor networks also achieve near to optimal rates in $O(n^{-s(\boldsymbol{\alpha})/d})$ with

$$s(\boldsymbol{\alpha})/d = (\alpha_1^{-1} + \ldots + \alpha_d^{-1})^{-1}$$

  the aggregated smoothness. Curse of dimensionality can be circumvented with sufficient anisotropy.
- Curse of dimensionality can be circumvented for non usual function classes such as compositions of smooth functions

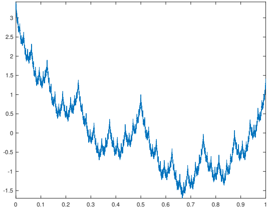
- Can approximate very efficiently functions with low (or no) regularity in a usual sense: discontinuous functions, nowhere differentiable functions, fractals

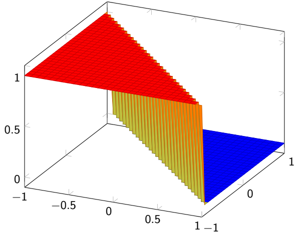


# Overview of results on approximation theory of tensor networks

- Can approximate very efficiently functions with low (or no) regularity in a usual sense: discontinuous functions, nowhere differentiable functions, fractals




- Approximation classes of tensor networks (using tensorization) are not embedded in any Besov space. Tensor networks can efficiently approximate functions beyond standard smoothness classes.

# Overview of results on approximation theory of tensor networks

- $\Phi_n^{\mathcal{F}}$ is closed while $\Phi_n^{\mathcal{S}}$ is not. Manipulating sparse tensor networks may be difficult in practice (to be compared with sparsely connected neural networks).

# Overview of results on approximation theory of tensor networks

- $\Phi_n^{\mathcal{F}}$ is closed while $\Phi_n^{\mathcal{S}}$ is not. Manipulating sparse tensor networks may be difficult in practice (to be compared with sparsely connected neural networks).
- Approximation classes of sparse tensor networks are larger than those of full tensor networks. However, if sparse tensor networks achieve a convergence in $O(n^{-\alpha})$, then full tensor networks achieve a convergence in $O(n^{-\alpha/2})$

# Overview of results on approximation theory of tensor networks

- $\Phi_n^{\mathcal{F}}$ is closed while $\Phi_n^{\mathcal{S}}$ is not. Manipulating sparse tensor networks may be difficult in practice (to be compared with sparsely connected neural networks).
- Approximation classes of sparse tensor networks are larger than those of full tensor networks. However, if sparse tensor networks achieve a convergence in $O(n^{-\alpha})$, then full tensor networks achieve a convergence in $O(n^{-\alpha/2})$
- The choice of the tensor format (ordering of variables, dimension partition tree $T$) may have a big influence on the performance.

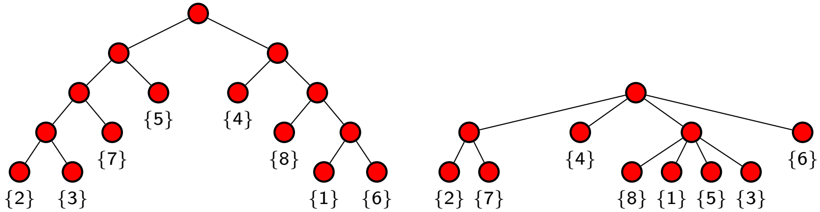
$$f(x) = f_1(x_1) f_{2|1}(x_2|x_1) \dots f_{d|d-1}(x_d|x_{d-1})$$

# Overview of results on approximation theory of tensor networks

- $\Phi_n^{\mathcal{F}}$ is closed while $\Phi_n^{\mathcal{S}}$ is not. Manipulating sparse tensor networks may be difficult in practice (to be compared with sparsely connected neural networks).
- Approximation classes of sparse tensor networks are larger than those of full tensor networks. However, if sparse tensor networks achieve a convergence in $O(n^{-\alpha})$, then full tensor networks achieve a convergence in $O(n^{-\alpha/2})$
- The choice of the tensor format (ordering of variables, dimension partition tree $T$) may have a big influence on the performance.

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

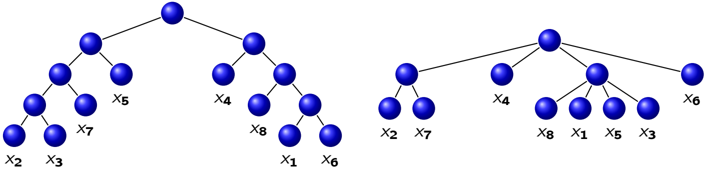
- How to select a good tree ? Combinatorial problem. Possible stochastic algorithms.



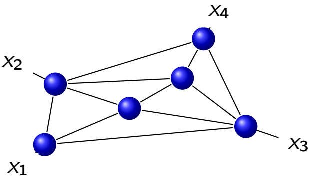
## Some open questions

- What are the properties of the approximation tool with free tree $T$ over $\{1, \dots, (L+1)d\}$

$$\Phi_n = \{f \in \Phi_{L,T,r,m} : L \in \mathbb{N}_0,\ T \subset 2^{\{1,\dots,(L+1)d\}}, r \in \mathbb{N}^{\#T}, compl(f) \le n\} \quad ?$$




- What about approximation classes of more general tensor networks ?



# Some open questions

- Algorithms to practically compute approximations achieving a certain precision with almost optimal complexity, using available information on the function (model equations, point samples...)
- Computational complexity of (deterministic or randomized) algorithms based on point samples for functions from approximation classes of tensor networks ?
- Theory to practice gap ?

**Approximation theory of neural networks**

R. Gribonval, G. Kutyniok, M. Nielsen, and F. Voigtlaender.
Approximation spaces of deep neural networks.
*Constructive approximation*, 55(1):259–367, 2022.

I. Gühring, M. Raslan, and G. Kutyniok.
Expressivity of deep neural networks.
*arXiv preprint arXiv:2007.04759*, 2020.

R. DeVore, B. Hanin, and G. Petrova.
Neural network approximation.
*Acta Numerica*, 30:327–444, 2021.

I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova.
Nonlinear approximation and (deep) ReLU networks.
*Constructive Approximation*, 55(1):127–172, 2022.

D. Yarotsky.
Error bounds for approximations with deep relu networks.
*Neural Networks*, 94:103–114, 2017.

# References II

D. Yarotsky and A. Zhevnerchuk.

The phase diagram of approximation rates for deep neural networks.

*Advances in neural information processing systems*, 33:13005–13015, 2020.

M. Ali and A. Nouy.

Approximation of smoothness classes by deep ReLU networks, arXiv:2007.15645, To appear in SIAM Journal on Numerical Analysis.

P. Grohs and F. Voigtländer.

Proof of the theory-to-practice gap in deep learning via sampling complexity bounds for neural network approximation spaces.

*CoRR*, abs/2104.02746, 2021.

**Introduction to tensors and tensor networks**

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.

R. Orus.

A practical introduction to tensor networks: Matrix product states and projected entangled pair states.

*Annals of Physics*, 349:117 – 158, 2014.

A. Falcó, W. Hackbusch, and A. Nouy.

Tree-based tensor formats.

*SeMA Journal*, Oct 2018.

W. Hackbusch.

Minimal divergence for border rank-2 tensor approximation.

*Linear and Multilinear Algebra*, pages 1–17, 2021.

**Approximation theory of tensor networks**

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.

M. Ali and A. Nouy.

Approximation with tensor networks. part i: Approximation spaces.

*ArXiv*, abs/2007.00118, 2020.

## References IV


M. Ali and A. Nouy.
Approximation with tensor networks. part ii: Approximation rates for smoothness classes.
*ArXiv*, abs/2007.00128, 2020.

M. Ali and A. Nouy.
Approximation with tensor networks. part iii: Multivariate approximation.
*arXiv preprint arXiv:2101.11932*, 2021.

M. Bachmayr, A. Nouy, and R. Schneider.
Approximation by tree tensor networks in high dimensions: Sobolev and compositional functions.
*arXiv preprint arXiv:2112.01474*, 2021.

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.

Valentin Khrulkov, Alexander Novikov, and Ivan Oseledets.
Expressive power of recurrent neural networks.
In *International Conference on Learning Representations*, 2018.

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

# References V

Vladimir Kazeev, Ivan Oseledets, Maxim Rakhuba, and Christoph Schwab.
Qtt-finite-element approximation for multiscale problems i: model problems in one dimension.
*Advances in Computational Mathematics*, 43(2):411–442, Apr 2017.