Méthodes d'approximation de faible rang pour la réduction de modèles

Anthony Nouy

Ecole Centrale Nantes, GeM

Joint work with Mathilde Chevreuil, Loic Giraldi, Prashant Rai

Problems in uncertainty quantification

Parameter-dependent models

$$\mathcal{M}(u(X); X) = 0$$

where $X = (X_1, \ldots, X_d)$ are random variables.

- Forward problem: evaluation of statistics, probability of events, sensitivity indices...

$$\mathbb{E}(f(u(X))) = \int_{\mathbb{R}^d} f(u(x_1, \ldots, x_d)) p(x_1, \ldots, x_d) dx_1 \ldots dx_d$$

- Inverse problem: from (partial) observations of u, estimate the density of X

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

Problems in uncertainty quantification

- Solving forward and inverse UQ problems requires the evaluation of the model for many instances of X.
- In practice, we rely on approximations of the solution map

$$x \mapsto u(x)$$

which are used as surrogate models.

Problems in uncertainty quantification

- Solving forward and inverse UQ problems requires the evaluation of the model for many instances of X.
- In practice, we rely on approximations of the solution map

$$x \mapsto u(x)$$

  which are used as surrogate models.
- Complexity issues:
  - High-dimensional functions

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

  - For complex numerical models, only a few evaluations of the function are available.

Problems in uncertainty quantification

- Solving forward and inverse UQ problems requires the evaluation of the model for many instances of X.
- In practice, we rely on approximations of the solution map

$$x \mapsto u(x)$$

  which are used as surrogate models.
- Complexity issues:
  - High-dimensional functions

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

  - For complex numerical models, only a few evaluations of the function are available.

The small data challenge!

High-dimensional approximation

The goal is to approximate a multivariate function

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

A "naive" tensor product approximation

$$u(x_1, \ldots, x_d) \approx \sum_{\alpha_1=1}^{n} \ldots \sum_{\alpha_d=1}^{n} a_{\alpha_1 \ldots \alpha_d} \psi_{\alpha_1}^{(1)}(x_1) \ldots \psi_{\alpha_d}^{(d)}(x_d), \quad a \in \mathbb{R}^{n^d}$$

yields a representation with a high number of parameters

$$N = n^d$$

High-dimensional approximation

The goal is to approximate a multivariate function

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

A "naive" tensor product approximation

$$u(x_1, \ldots, x_d) \approx \sum_{\alpha_1=1}^{n} \ldots \sum_{\alpha_d=1}^{n} a_{\alpha_1 \ldots \alpha_d} \psi_{\alpha_1}^{(1)}(x_1) \ldots \psi_{\alpha_d}^{(d)}(x_d), \quad a \in \mathbb{R}^{n^d}$$

yields a representation with a high number of parameters

$$N = n^d$$

- Assuming $u \in C^s$, the number of parameters to achieve a precision ϵ is

$$N = O(\epsilon^{-d/s}) \quad \text{(Curse of dimensionality)}$$

High-dimensional approximation

The goal is to approximate a multivariate function

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

A "naive" tensor product approximation

$$u(x_1, \ldots, x_d) \approx \sum_{\alpha_1=1}^{n} \ldots \sum_{\alpha_d=1}^{n} a_{\alpha_1 \ldots \alpha_d} \psi_{\alpha_1}^{(1)}(x_1) \ldots \psi_{\alpha_d}^{(d)}(x_d), \quad a \in \mathbb{R}^{n^d}$$

yields a representation with a high number of parameters

$$N = n^d$$

- Assuming $u \in C^s$, the number of parameters to achieve a precision ϵ is

$$N = O(\epsilon^{-d/s}) \quad \text{(Curse of dimensionality)}$$

- Extra smoothness does not help.

How to beat the curse of dimensionality ?

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).

Approximations are searched in subsets M_n with a number of parameters

$$n = O(d^p)$$

but

- M_n is usually nonlinear, and
- M_n may be non smooth.

This turns approximation problems

$$\min_{v \in M_n} d(u, v)$$

into nonlinear and possibly non smooth optimization problems.

Low-dimensional models for high-dimensional approximation

- Low-order interactions, e.g.
  - No interaction (additive model)

$$u(x_1, \ldots, x_d) \approx u_0 + u_1(x_1) + \ldots + u_d(x_d)$$

  - First-order interactions

$$u(x_1, \ldots, x_d) \approx u_0 + \sum_i u_i(x_i) + \sum_{i \neq j} u_{i,j}(x_i, x_j)$$

- Small number of interactions
  - For a given $\Lambda \subset 2^{\{1,\ldots,d\}}$ (set of interaction groups),

$$u(x_1, \ldots, x_d) \approx \sum_{\alpha \in \Lambda} u_\alpha(x_\alpha)$$

  - Λ as a parameter

$$u(x_1, \ldots, x_d) \approx \sum_{\alpha \in \Lambda} u_\alpha(x_\alpha) \quad \text{with} \quad \#\Lambda = n$$

Low-dimensional models for high-dimensional approximation

- Sparsity relatively to a basis or frame $\{\psi_\alpha\}_{\alpha\in\mathbb{N}}$

$$u(x_1,\ldots,x_d) \approx \sum_{\alpha\in\Lambda} a_\alpha \psi_\alpha(x_1,\ldots,x_d), \quad \#\Lambda = n$$

- Sparsity relatively to a dictionary $\mathcal{D}$

$$u(x_1,\ldots,x_d) \approx \sum_{i=1}^{n} a_i \psi_i(x_1,\ldots,x_d), \quad \psi_i \in \mathcal{D}$$

Low-dimensional models for high-dimensional approximation

- Low rank,

$$u(x_1, \ldots, x_d) \approx u_1(x_1) \ldots u_d(x_d)$$

$$u(x_1, \ldots, x_d) \approx \sum_{i=1}^{r} u_{1,i}(x_1) \ldots u_{d,i}(x_d)$$

$$u(x_1, \ldots, x_d) \approx \sum_{i_1=1}^{r_1} \ldots \sum_{i_{d-1}=1}^{r_{d-1}} u_{1,i_1}(x_1) u_{i_1,i_2}(x_2) \ldots u_{i_{d-1},1}(x_d)$$

$$\ldots$$

Outline

1. Rank-structured approximation
2. Statistical learning methods for tensor approximation
3. Adaptive approximation in tree-based low-rank formats

Outline

1. Rank-structured approximation
2. Statistical learning methods for tensor approximation
3. Adaptive approximation in tree-based low-rank formats

Tensor product of functions

For $1 \leq \nu \leq d$, let V_ν be a space of functions defined on an interval $\mathcal{X}_\nu \subset \mathbb{R}$.

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

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

For example, for $i \in \mathbb{N}_0^d$, the monomial $x^i = x_1^{i_1} \ldots x_d^{i_d}$ is an elementary tensor.

Tensor product of functions

The algebraic tensor product of spaces V_ν, denoted

$$V_1 \otimes \ldots \otimes V_d,$$

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

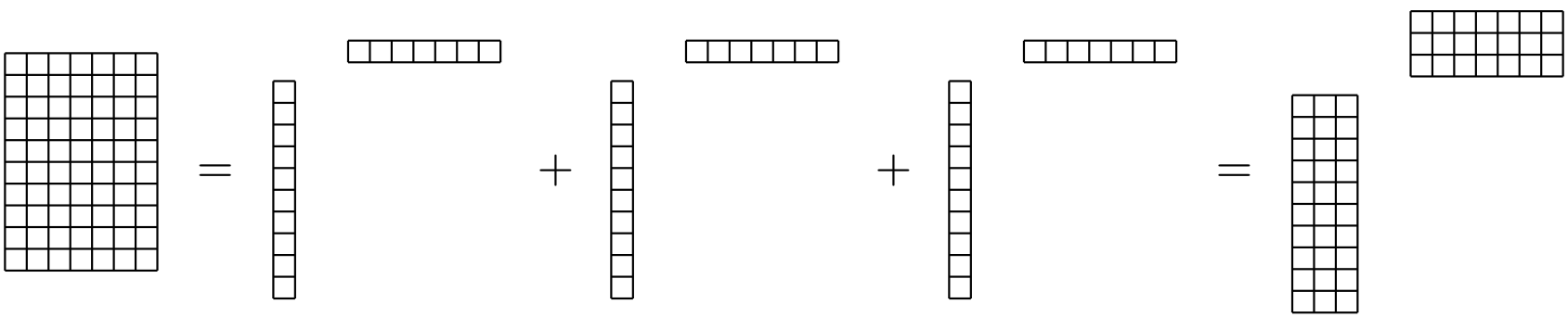
$$v(x) = \sum_{k=1}^{n} v_k^{(1)}(x_1) \ldots v_k^{(d)}(x_d).$$

Rank of order-two tensor

In the case $d = 2$, the rank of a bivariate function $v(x_1, x_2)$ is defined as the minimal integer r such that

$$v(x_1, x_2) = \sum_{i=1}^{r} v_i^{(1)}(x_1) v_i^{(2)}(x_2).$$

For a matrix $v \in \mathbb{R}^{n \times m}$ representing the values of a function on a grid, of the coefficients of the function on a product basis, this corresponds to the standard notion of matrix rank.



The set of order-two tensors with rank bounded by r, although it is not a linear space nor a convex set, has many favorable properties for numerical use.

Canonical rank of higher-order tensors

For tensors $v \in V_1 \otimes \ldots \otimes V_d$ with $d \geq 3$, there are different notions of rank.

The canonical rank of a function $v(x_1, \ldots, x_d)$, which is the natural extension of the notion of rank for order-two tensors, is the minimal integer r such that

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

where the $v_k^{(\nu)}(x_\nu)$ are in the function space V_ν.

The storage complexity of such a function is

$$\text{storage}(v) = r \sum_{\nu=1}^{d} \dim(V_\nu) = O(rdn),$$

for $\dim(V_\nu) = O(n)$, and scales linearly in d.

Canonical rank of higher-order tensors

But for $d \geq 3$, the set of tensors with canonical rank bounded by r looses many of the favorable properties of the case $d = 2$. The consequence is that no robust numerical method exists for approximation in this format.

Matricisation of tensors

For a non-empty subset α of $D = \{1, \ldots, d\}$, an order-d tensor $u(x_1, \ldots, x_d)$ can be identified with an order-two tensor

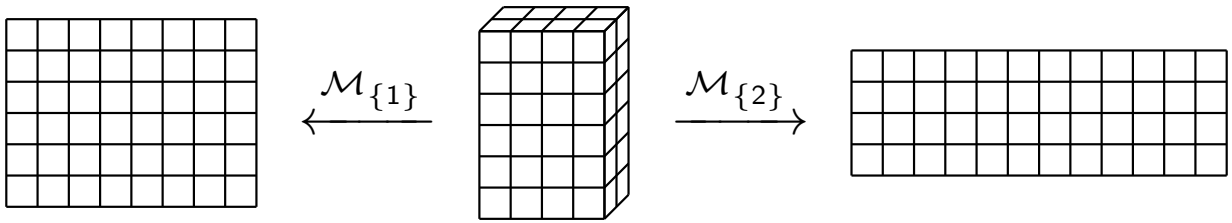
$$\mathcal{M}_\alpha(u)(x) = u(x_\alpha, x_{\alpha^c}),$$

where

$$x_\alpha = \{x_\nu\}_{\nu \in \alpha} \quad \text{and} \quad x_{\alpha^c} = \{x_\nu\}_{\nu \in \alpha^c}$$

are two complementary groups of variables.

The operator $\mathcal{M}_\alpha$ is called a matricisation operator.



α-rank

The α-rank of u, denoted $\text{rank}_\alpha(u)$, is the rank of the order-two tensor $\mathcal{M}_\alpha(u)$,

$$\text{rank}_\alpha(u) = \text{rank}(\mathcal{M}_\alpha(u)).$$

The motivation behind the definition of approximation formats based on α-ranks is to benefit from the nice properties of the rank of order-two tensors.

A multivariate function $u(x_1, \ldots, x_d)$ with $\text{rank}_\alpha(u) \leq r_\alpha$ is such that

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

Example

$u(x_1, \ldots, x_d) = u_1(x_1) + \ldots + u_d(x_d)$ where $u_1, \ldots, u_d$ are non constant functions satisfies $\text{rank}_\alpha(u) = 2$ for all α.

The corresponding storage complexity is

$$\text{storage}(v) = O(r_\alpha(n^{\#\alpha} + n^{\#\alpha^c})).$$

α-ranks and related tensor formats

For T a collection of subsets of D, we define the T-rank of a tensor v, denoted $\text{rank}_T(u)$, as the tuple

$$\text{rank}_T(v) = \{\text{rank}_\alpha(v)\}_{\alpha \in T}.$$

The subset of tensors in V with T-rank bounded by $r = (r_\alpha)_{\alpha \in T}$ is

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

As a finite intersection of subsets of tensors with bounded α-ranks, $\mathcal{T}_r^T$ inherits from geometrical and topological properties which are favorable for numerical simulation.

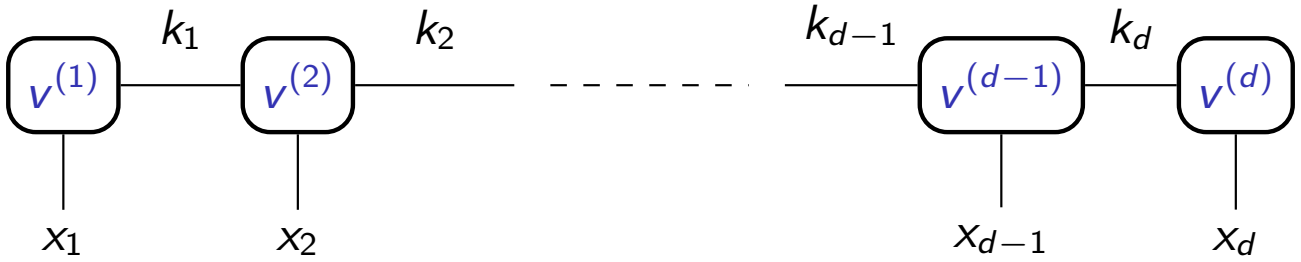
Tensor train format

For

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

$\text{rank}_T(v)$ is called the TT-rank of the tensor v. A tensor v with $\text{rank}_T(v) \leq r = (r_1, \ldots, r_{d-1})$ admits a representation

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



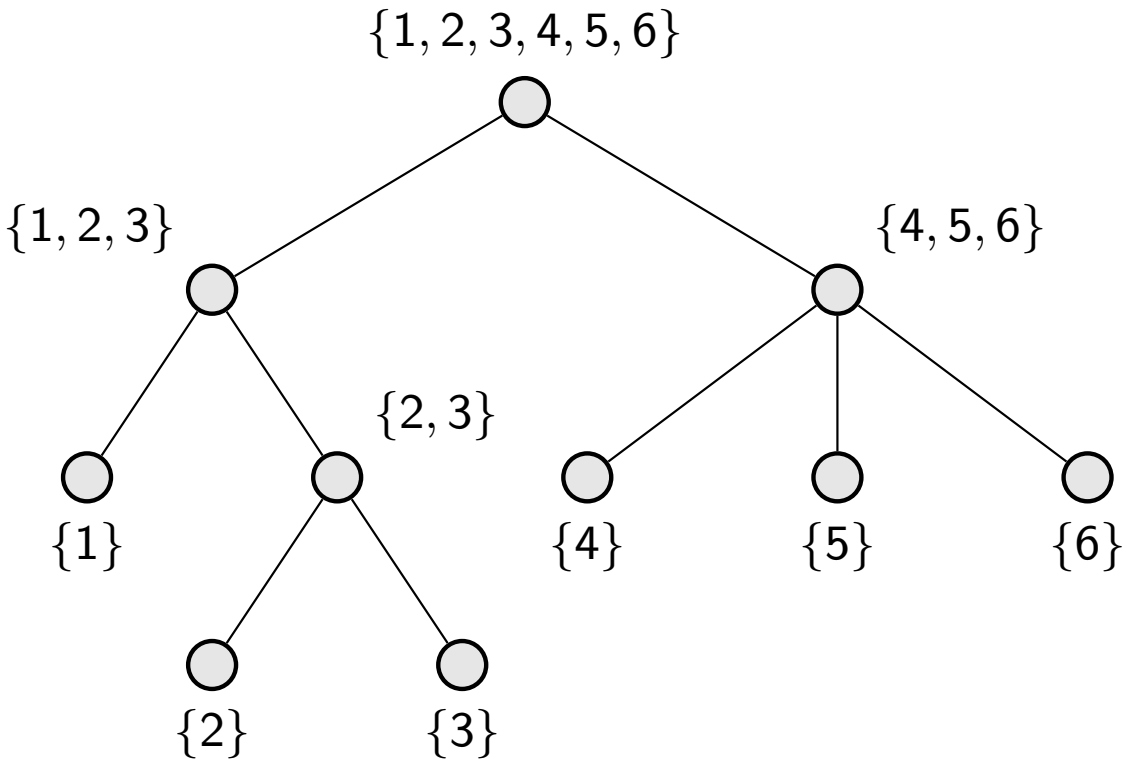
The storage complexity of an element in $\mathcal{TT}_r$ is

$$\text{storage}(v) = \sum_{\nu=1}^{d} r_{\nu-1} r_\nu \dim(V_\nu) = O(dnR^2)$$

with $\dim(V_\nu) = O(n)$, $r_\nu = O(R)$.

Tree-based (hierarchical) Tucker format

Tree-based (or hierarchical) Tucker formats are associated with a partition dimension tree T over $D = \{1, \ldots, d\}$, with root D and leaves $\{\nu\}$, $1 \leq \nu \leq d$.



The tree-based rank of a tensor v is the tuple $\text{rank}_T(v) = (\text{rank}_\alpha(v))_{\alpha \in T}$.

Parametrization and storage of low-rank tensor formats

Ultimately, a tensor in a certain low-rank tensor format $\mathcal{M}_{\leq r} = \{v : \text{rank}(v) \leq r\}$ admits a multilinear parametrization of the form

$$v(x) = \sum_{k_1=1}^{r_1} \cdots \sum_{k_L=1}^{r_L} \prod_{\nu=1}^{d} p_\nu\left(x_\nu, (k_i)_{i\in s_\nu}\right) \prod_{\nu=d+1}^{M} p_\nu\left((k_i)_{i\in s_\nu}\right) = \Psi(p_1, \ldots, p_M)(x)$$

where the parameter p_ν is a tensor which depends on a small subset of summation variables $(k_i)_{i\in s_\nu}$.

The storage complexity scales as

$$O(dnR^k)$$

where R is a bound of the representation ranks.

Outline

1. Rank-structured approximation
2. Statistical learning methods for tensor approximation
3. Adaptive approximation in tree-based low-rank formats

Statistical learning methods for tensor approximation

- Approximation of a function $u(X) = u(X_1, \ldots, X_d)$ from evaluations $\{y_k = u(x^k)\}_{k=1}^K$ on a training set $\{x^k\}_{k=1}^K$ (i.i.d. samples of X)

Statistical learning methods for tensor approximation

- Approximation of a function $u(X) = u(X_1, \ldots, X_d)$ from evaluations $\{y_k = u(x^k)\}_{k=1}^K$ on a training set $\{x^k\}_{k=1}^K$ (i.i.d. samples of X)
- Approximation in subsets of rank-structured functions

$$\mathcal{M}_{\leq r} = \{v : rank(v) \leq r\}$$

by minimization of an empirical risk

$$\widehat{\mathcal{R}}_K(v) = \frac{1}{K} \sum_{k=1}^{K} \ell(u(x^k), v(x^k))$$

where ℓ is a certain loss function.

Statistical learning methods for tensor approximation

- Approximation of a function $u(X) = u(X_1, \ldots, X_d)$ from evaluations $\{y_k = u(x^k)\}_{k=1}^K$ on a training set $\{x^k\}_{k=1}^K$ (i.i.d. samples of X)
- Approximation in subsets of rank-structured functions

$$\mathcal{M}_{\leq r} = \{v : rank(v) \leq r\}$$

by minimization of an empirical risk

$$\widehat{\mathcal{R}}_K(v) = \frac{1}{K} \sum_{k=1}^K \ell(u(x^k), v(x^k))$$

where ℓ is a certain loss function.

- Here, we consider for least-squares regression

$$\widehat{\mathcal{R}}_K(v) = \frac{1}{K} \sum_{k=1}^K (u(x^k) - v(x^k))^2 = \widehat{\mathbb{E}}_K((u(X) - v(X))^2)$$

but other loss functions could be used for different objectives than L^2-approximation (e.g. classification).

Alternating minimization algorithm

- Multilinear parametrization of tensor manifolds

$$\mathcal{M}_{\leq r} = \{v = \Psi(p_1, \ldots, p_L) : p_l \in \mathbb{R}^{m_l}, 1 \leq l \leq L\}$$

so that

$$\min_{v \in \mathcal{M}_{\leq r}} \widehat{\mathcal{R}}_K(v) = \min_{p_1, \ldots, p_M} \widehat{\mathcal{R}}_K(\Psi(p_1, \ldots, p_M))$$

Alternating minimization algorithm

- Multilinear parametrization of tensor manifolds

$$\mathcal{M}_{\leq r} = \{v = \Psi(p_1, \ldots, p_L) : p_l \in \mathbb{R}^{m_l}, 1 \leq l \leq L\}$$

so that

$$\min_{v \in \mathcal{M}_{\leq r}} \widehat{\mathcal{R}}_K(v) = \min_{p_1, \ldots, p_M} \widehat{\mathcal{R}}_K(\Psi(p_1, \ldots, p_M))$$

- Alternating minimization algorithm: Successive minimization problems

$$\min_{p_l \in \mathbb{R}^{m_l}} \widehat{\mathcal{R}}_K(\underbrace{\Psi(p_1, \ldots, p_l, \ldots, p_M)}_{\Psi_l(\cdot)^T p_l})$$

which are standard linear approximation problems

$$\min_{p_l \in \mathbb{R}^{m_l}} \frac{1}{K} \sum_{k=1}^{K} \ell(u(x^k), \Psi_l(x^k)^T p_l)$$

Regularization

- Regularization

$$\min_{p_1,\ldots,p_M} \widehat{\mathcal{R}}_K(\Psi(p_1,\ldots,p_M)) + \sum_{l=1}^{L} \lambda_l \Omega_l(p_l)$$

  with regularization functionals Ω_l promoting
  - sparsity (e.g. $\Omega_l(p_l) = \|p_l\|_1$),
  - smoothness,
  - ...
- Alternating minimization algorithm requires the solution of successive standard regularized linear approximation problems

$$\min_{p_l} \frac{1}{K} \sum_{k=1}^{K} \ell(u(x^k), \Psi_l(x^k)^T p_l) + \lambda_l \Omega_l(p_l) \qquad (\star)$$

  - For square-loss and $\Omega_l(p_l) = \|p_l\|_1$, $(\star)$ is a LASSO problem.
- Cross-validation methods for the selection of regularization parameters λ_l.

Illustrations

- Approximations in canonical format

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

  or in tensor-train (TT) format:

$$v(x_1, \ldots, x_d) = \sum_{k_1=1}^{r_1} \ldots \sum_{k_{d-1}=1}^{r_{d-1}} v_{1,k_1}^{(1)}(x_1) \ldots v_{k_{d-1},1}^{(d)}(x_d)$$

- Polynomial approximations

$$v_k^{(\nu)} \quad \text{and} \quad v_{i_{k-1},i_k}^{(\nu)} \in \mathbb{P}_q$$

- $v = \Psi(p_1, \ldots, p_d)$ with parameter p_ν gathering the coefficients of functions of x_ν on a polynomial basis (orthonormal in $L^2_{P_{X_\nu}}(\mathcal{X}_\nu)$).
- Sparsity inducing regularization and cross-validation (leave one out) for the automatic selection of polynomial basis functions. Use of standard least-squares in the selected basis.

Illustration : 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)	$U(9855, 12045)$

- Polynomial approximation with degree $q = 8$.
- Test set of size 1000.

Illustration : Borehole function

- Approximation in canonical format for different sizes K of training set, selection of optimal rank.

K	rank	test error
100	2	$1.0\,10^{-3}$
1000	5	$3.8\,10^{-4}$
10000	7	$6.0\,10^{-6}$

Illustration : Borehole function

- Approximation in canonical format for different sizes K of training set, selection of optimal rank.

K	rank	test error
100	2	$1.0\,10^{-3}$
1000	5	$3.8\,10^{-4}$
10000	7	$6.0\,10^{-6}$

- Approximation in tensor train format for different ranks and for different sizes K of the training set.

rank	K = 100	K=1000	K=10000
(1 1 1 1 1 1 1)	$1.7\,10^{-2}$	$1.4\,10^{-2}$	$1.4\,10^{-2}$
(2 2 2 2 2 2 2)	$6.7\,10^{-4}$	$9.1\,10^{-4}$	$3.3\,10^{-4}$
(3 3 3 3 3 3 3)	$3.2\,10^{-3}$	$1.2\,10^{-4}$	$1.0\,10^{-5}$
(4 4 4 4 4 4 4)	$2.1\,10^{-1}$	$7.6\,10^{-5}$	$1.9\,10^{-7}$
(5 5 5 5 5 5 5)	$7.3\,10^{0}$	$3.8\,10^{-4}$	$2.8\,10^{-7}$
(6 6 6 6 6 6 6)	$7.9\,10^{-1}$	$4.1\,10^{-3}$	$2.1\,10^{-7}$

Illustration : Borehole function

- Approximation in canonical format for different sizes K of training set, selection of optimal rank.

K	rank	test error
100	2	$1.0\,10^{-3}$
1000	5	$3.8\,10^{-4}$
10000	7	$6.0\,10^{-6}$

- Approximation in tensor train format for different ranks and for different sizes K of the training set.

rank	K = 100	K=1000	K=10000
(1 1 1 1 1 1 1)	$1.7\,10^{-2}$	$1.4\,10^{-2}$	$1.4\,10^{-2}$
(2 2 2 2 2 2 2)	$6.7\,10^{-4}$	$9.1\,10^{-4}$	$3.3\,10^{-4}$
(3 3 3 3 3 3 3)	$3.2\,10^{-3}$	$1.2\,10^{-4}$	$1.0\,10^{-5}$
(4 4 4 4 4 4 4)	$2.1\,10^{-1}$	$7.6\,10^{-5}$	$1.9\,10^{-7}$
(5 5 5 5 5 5 5)	$7.3\,10^{0}$	$3.8\,10^{-4}$	$2.8\,10^{-7}$
(6 6 6 6 6 6 6)	$7.9\,10^{-1}$	$4.1\,10^{-3}$	$2.1\,10^{-7}$

Finding optimal rank is a combinatorial problem...

Outline

1. Rank-structured approximation
2. Statistical learning methods for tensor approximation
3. Adaptive approximation in tree-based low-rank formats

Heuristic strategy for rank adaptation (tree-based Tucker format)

- Given $T \subset 2^{\{1,\dots,d\}}$, construction of a sequence of approximations u_m in tree-based Tucker format with increasing rank:

$$u_m \in \{v : rank_T(v) \leq (r_\alpha^m)_{\alpha \in T}\}$$

Heuristic strategy for rank adaptation (tree-based Tucker format)

- Given $T \subset 2^{\{1,\dots,d\}}$, construction of a sequence of approximations u_m in tree-based Tucker format with increasing rank:

$$u_m \in \{v : rank_T(v) \leq (r_\alpha^m)_{\alpha \in T}\}$$

- At iteration m,

$$\begin{cases} r_\alpha^{m+1} = r_\alpha^m + 1 & \text{if } \alpha \in T_m \\ r_\alpha^{m+1} = r_\alpha^m & \text{if } \alpha \notin T_m \end{cases}$$

where T_m is selected in order to obtain (hopefully) the fastest decrease of the error.

Heuristic strategy for rank adaptation (tree-based Tucker format)

- Given $T \subset 2^{\{1,\ldots,d\}}$, construction of a sequence of approximations u_m in tree-based Tucker format with increasing rank:

$$u_m \in \{v : rank_T(v) \leq (r_\alpha^m)_{\alpha \in T}\}$$

- At iteration m,

$$\begin{cases} r_\alpha^{m+1} = r_\alpha^m + 1 & \text{if } \alpha \in T_m \\ r_\alpha^{m+1} = r_\alpha^m & \text{if } \alpha \notin T_m \end{cases}$$

  where T_m is selected in order to obtain (hopefully) the fastest decrease of the error.

- A possible strategy consists in computing the singular values

$$\sigma_1^\alpha \geq \ldots \geq \sigma_{r_\alpha^m}^\alpha$$

  of α-matricizations $\mathcal{M}_\alpha(u_m)$ of u_m for all $\alpha \in T$. Letting $0 \leq \theta \leq 1$, can we choose

$$T_m = \left\{ \alpha \in T : \sigma_{r_\alpha^m}^\alpha \geq \theta \max_{\beta \in T} \sigma_{r_\beta^m}^\beta \right\}$$

Illustration : Borehole function

- Training set of size $K = 1000$

iteration	rank	test error
0	(1 1 1 1 1 1 1)	$1.4\,10^{-2}$
1	(2 2 2 2 2 2 2)	$4.4\,10^{-4}$
2	(2 2 2 3 3 2 2)	$8.1\,10^{-6}$
3	(3 3 3 4 3 2 2)	$6.2\,10^{-6}$
4	(3 3 3 4 4 3 2)	$2.1\,10^{-5}$
5	(3 3 3 4 4 3 3)	$9.6\,10^{-6}$
6	(3 4 4 4 5 4 4)	$1.5\,10^{-5}$

The selected rank is one order of magnitude better than the optimal "isotropic" rank $(r, r, \ldots, r)$

Illustration : Borehole function

- Different sizes K of training set, selection of optimal ranks.

TT format

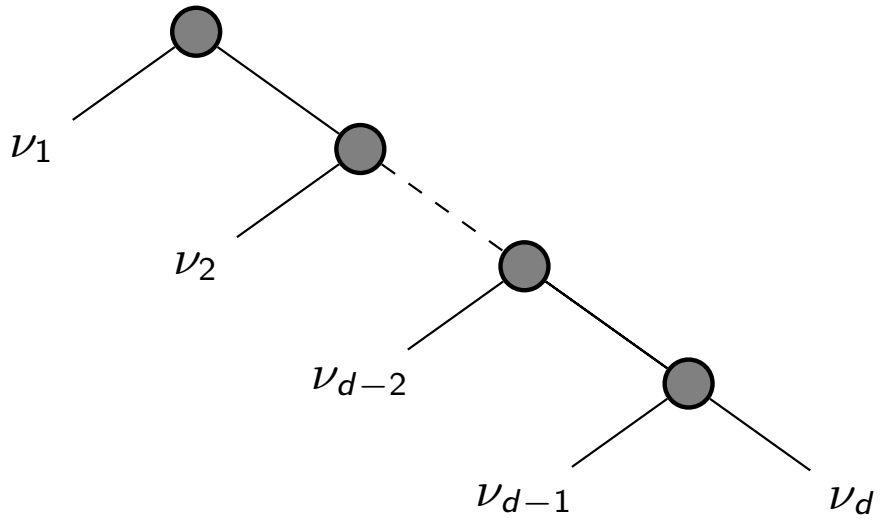
K	rank	test error
100	(3 4 4 3 3 2 1)	$7.1\,10^{-4}$
1000	(3 3 3 4 4 3 2)	$6.2\,10^{-6}$
10000	(5 6 6 7 7 5 4)	$2.4\,10^{-8}$

Canonical format

K	rank	test error
100	2	$1.0\,10^{-3}$
1000	5	$3.8\,10^{-4}$
10000	7	$6.0\,10^{-6}$

Influence of the tree (TT format)

- Test error for different trees T (Training set of size $K = 50$)



tree	$\{\nu_1, \ldots, \nu_d\}$	optimal rank	test error
T_1	(1 2 3 4 5 6 7 8)	(2 2 2 2 2 1 1)	$6.2\,10^{-4}$
T_2	(1 3 8 5 6 2 4 7)	(2 2 2 2 2 2 1)	$1.3\,10^{-3}$
T_3	(7 6 8 1 4 5 2 3)	(1 1 1 1 1 1 1)	$1.5\,10^{-2}$
T_4	(8 2 4 7 5 1 3 6)	(1 1 2 3 3 2 2)	$1.3\,10^{-2}$

Finding the optimal tree is a combinatorial problem...

Strategy for tree adaptation in (TT format)

Starting from an initial tree, we perform iteratively the following two steps:

- Run the algorithm with rank adaptation to compute an approximation v associated with the current tree

$$v(x_1, \ldots, x_d) = \sum_{i_1=1}^{r_1} \cdots \sum_{i_{d-1}=1}^{r_{d-1}} v^1_{1,i_1}(x_{\nu_1}) \ldots v^d_{i_{d-1},1}(x_{\nu_d})$$

- Run a tree optimization algorithm yielding an equivalent representation of v (at the current precision)

$$v(x_1, \ldots, x_d) \approx \sum_{i_1=1}^{\tilde{r}_1} \cdots \sum_{i_{d-1}=1}^{\tilde{r}_{d-1}} v^1_{1,i_1}(x_{\tilde{\nu}_1}) \ldots v^d_{i_{d-1},1}(x_{\tilde{\nu}_d})$$

with reduced ranks $\{\tilde{r}_1, \ldots, \tilde{r}_{d-1}\}$, where $\{\tilde{\nu}_1, \ldots, \nu_d\}$ is a permutation of $\{\nu_1, \ldots, \nu_d\}$.

Strategy for tree adaptation in (TT format)

Illustration with training set of size $K = 50$.
We run the algorithm for different initial trees.
Indicated in blue are the permuted dimensions in the final tree.

tree	$\{\nu_1, \ldots, \nu_d\}$	optimal rank	test error
initial	(1 2 3 4 5 6 7 8)	(2 2 2 2 2 1 1)	$6.2\,10^{-4}$
final	(1 2 3 5 4 6 7 8)	(2 2 2 2 2 1 1)	$4.5\,10^{-4}$
initial	(1 3 8 5 6 2 4 7)	(2 2 2 2 2 2 1)	$1.3\,10^{-3}$
final	(1 3 8 5 2 6 4 7)	(2 2 2 2 2 2 1)	$5.1\,10^{-4}$
initial and final	(7 6 8 1 4 5 2 3)	(1 1 1 1 1 1 1)	$1.5\,10^{-2}$
initial	(8 2 4 7 5 1 3 6)	(1 1 2 3 3 2 2)	$1.3\,10^{-2}$
	(8 2 7 5 1 4 3 6)	(1 1 2 2 2 2 2)	$1.2\,10^{-3}$
final	(8 2 7 5 1 3 4 6)	(1 1 2 2 2 2 2)	$1.3\,10^{-3}$

Concluding remarks and on-going works

- Need for robust strategies for tree adaptation.
- "Statistical dimension" of low-rank subsets ?
- Adaptive sampling strategies.
- Goal-oriented construction of low-rank approximations.

Concluding remarks and on-going works

- Need for robust strategies for tree adaptation.
- "Statistical dimension" of low-rank subsets ?
- Adaptive sampling strategies.
- Goal-oriented construction of low-rank approximations.
- For Régis, estimation of high-dimensional copula

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

using low-rank formats.

References

A. Nouy.
Low-rank methods for high-dimensional approximation and model order reduction.
arXiv:1511.01554

Anthony Nouy.
Low-Rank Tensor Methods for Model Order Reduction, pages 1–26.
Springer International Publishing, Cham, 2016.

M. Chevreuil, R. Lebrun, A. Nouy, and P. Rai.
A least-squares method for sparse low rank approximation of multivariate functions.
SIAM/ASA Journal on Uncertainty Quantification, 3(1):897–921, 2015.