A new method for the reconstruction of multivariate functions by using the Tucker decomposition

Application to the reconstruction of cross-sections in neutronic

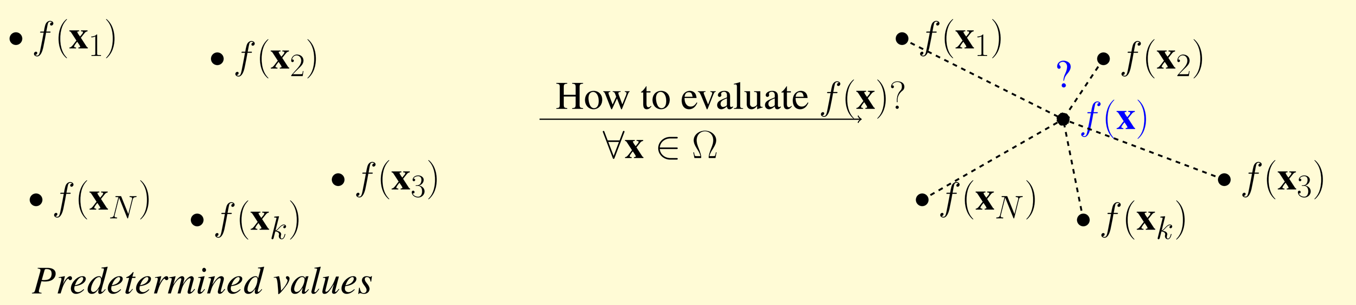
Thi Hieu LUU, Yvon MADAY, Matthieu GUILLO, Pierre GUÉRIN

Department of SINETICS, EDF-R&D and *Jacques-Louis Lions Laboratory, UPMC, France*

Reconstruction of multivariate functions

$f : \mathbf{x} = (x_1, \ldots, x_d) \in \Omega \subset \mathbb{R}^d \longmapsto f(\mathbf{x}) \in \mathbb{R}$

$\{\mathbf{x}_k\}_{k=1}^N \subset \Omega$ and $\{f(\mathbf{x})\}_{k=1}^N$ are predetermined values of f on the set $\{\mathbf{x}_k\}_{k=1}^N$



A classical method such as *the multilinear interpolation* has the complexity of the order of $\mathcal{O}(n^d)$, due to the number of points in its discretization as well as its storage size.

$\Longrightarrow$ ***Curse of dimensionality problem***.

$\Longrightarrow$ ***Require a new technique for the reconstruction of multivariate functions***.

Tucker decomposition method

Tucker decomposition [1], [2], [3]:

$$\underbrace{f(x_1, \ldots, x_d)}_{\text{multivariate functions}} \approx \tilde{f}(x_1, \ldots, x_d) = \sum_{i_1=1}^{r_1} \cdots \sum_{i_d=1}^{r_d} \mathbf{a}[i_1, \ldots, i_d] \prod_{j=1}^{d} \underbrace{\varphi_{i_j}^{(j)}(x_j)}_{\text{one-variate function}}$$

Need to be determined:

1. The ***basis functions*** $\{\varphi_{i_j}^{(j)}(x_j)\}_{i_j=1}^{r_j}$ for each direction j, $1 \le j \le d$.
2. The ***coefficients*** $\mathbf{a_i} = \mathbf{a}[i_1, \ldots, i_d]$, $1 \le \mathbf{i} \le R$ with $R = \prod_{j=1}^d r_j \overset{r_j = r}{=} \mathcal{O}(r^d)$.

$\Longrightarrow$ ***The number of predetermined values and the size of stored data are scaled from*** $\mathcal{O}(n^d)$ ***to*** $\mathcal{O}(r^d)$, where r is often small and $r \ll n$ in general.

1. Determination of the *basis functions* by using the *Karhunen-Loève decomposition*

For one ***concerned direction*** *j:*

- Setting $x = x_j$
  $\hookrightarrow$ **Fine discretization of the j^{th}-axis into n_j points ($n_j \approx 10$).**
- Condensing the remaining parameters into one parameter **y**: $\mathbf{y} := (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d)$
  $\hookrightarrow$ **Coarse discretization for the other directions (for instance: 2 points).**
- Solving numerically an eigenvalue problem obtained from the Karhunen-Loève decomposition, for $x = x_j$:

$$A\varphi^{(x)} = \lambda\varphi^{(x)}, \text{ where } A\varphi^{(x)} \leftrightarrow \int_{\Omega_x}\int_{\Omega_\mathbf{y}} f(x, \mathbf{y}) f(x', \mathbf{y}) \varphi^{(x)}(x') dx' d\mathbf{y} \quad (1)$$

Solution: $\{\varphi_{i_j}^{(x)}\}_{i_j=1}^{r_j}$ which correspond to the r_j largest eigenvalues are chosen as the basis functions for the direction j.

2. Determination of the coefficients

- Solving a system of linear equations for R unknown coefficients **a** with $R = \prod_{j=1}^d r_j$:

$$\begin{cases} f(\mathbf{x}_1) = \sum_{\mathbf{i}=1}^R \mathbf{a_i} \prod_{j=1}^d \varphi_{i_j}^{(j)}(x_{1_j}) \\ \ldots \\ f(\mathbf{x}_t) = \sum_{\mathbf{i}=1}^R \mathbf{a_i} \prod_{j=1}^d \varphi_{i_j}^{(j)}(x_{t_j}) \\ \ldots \\ f(\mathbf{x}_R) = \sum_{\mathbf{i}=1}^R \mathbf{a_i} \prod_{j=1}^d \varphi_{i_j}^{(j)}(x_{R_j}) \end{cases} \quad (2)$$

$\Longrightarrow$ ***Require a set of R points*** $\{\mathbf{x}_t\}_{t=1}^R$ ***on the left hand side of (2).***

Solution: $\{\mathbf{x}_t\}_{t=1}^R$ is proposed as a tensor product of the sets $\{x_{t_j}^{(j)}\}_{t_j=1}^{r_j}$:

$$\{\mathbf{x}_t\}_{t=1}^R = \otimes_{j=1}^d \{x_{t_j}^{(j)}\}_{t_j=1}^{r_j} \quad (3)$$

where $\{x_{t_j}^{(j)}\}_{t_j=1}^{r_j}$ are selected by applying greedy algorithm [4] to $\{\varphi_{i_j}^{(j)}\}_{i_j=1}^{r_j}$ and $\{x_1^{(j)}, \ldots, x_{n_j}^{(j)}\}$ - the fine discretized points of the j^{th}-direction.

3. Results improvement with sparse representation for the coefficients

- *A posteriori* sparse representation ($\searrow$ storage for $\{\mathbf{a}\}$): after determining the coefficients by the system (2), the smallest coefficients $\mathbf{a_i}$ are eliminated if their values satisfy, e.g :

$$\frac{|\mathbf{a_i}|}{\sum_\mathbf{i} |\mathbf{a_i}|} < \epsilon, \text{ for a chosen } \epsilon \quad (4)$$

- *A priori* sparse representation ($\searrow$ predetermined values by (2), $\searrow$ storage for $\{\mathbf{a}\}$): before determining the coefficients by the system (2), the coefficients $\mathbf{a_i}$ are eliminated if their indexes satisfy, e.g :

$$\prod_{j=1}^d \frac{i_j}{r_j} > \epsilon, \text{ for a chosen } \epsilon \quad (5)$$

Application to the reconstruction of cross-sections in neutronic

Cross-sections $\Sigma_i \Leftrightarrow$ multivariate functions

- Depend locally on various parameters which characterize the material behavior of a nuclear reactor, for instance: *Burnup, fuel temperature, moderator density, boron concentration and xenon level, etc.*:

$$\text{cross-sections} \rightsquigarrow \Sigma_i \underbrace{(Bu, T_c, \rho_m, C_b, Xenon, \ldots)}_{\mathbf{x} \in \Omega \subset \mathbb{R}^d,\, d \ge 5}$$

Reconstruction of cross-sections problem

- Reconstruction applied not to one cross-section but to a set of cross-sections $\{\Sigma_i\}_i$ over Ω.
- Reduce as much as possible the predetermined data and the storage size, compared to the multilinear model.
- Ensure the accuracy.

Results of a benchmark [3], [5]: cross-sections depend on five parameters

Model	Number of predetermined values	Remark
Tucker	**1098**	***The Tucker decomposition reduces the number of predetermined values by 38%.***
Multilinear	**1782**	

Table 1: Comparison of the number of predetermined values

Cross - section	Storage (number of floats) Tucker	Multilinear	Remark
Σ_1	**244**	**1782**	***The Tucker decomposition reduces the size of stored data by a factor from 4 to 9.***
Σ_2	**192**	**1782**	
Σ_3	**244**	**1782**	
Σ_4	**371**	**1782**	
Σ_5	**416**	**1782**	
Σ_6	**192**	**1782**	

Table 2: Comparison of the stored data size

$$\mathbf{e}^{(\mathbf{inf})} = \max_{\mathbf{x} \in \textit{reference grid}} \frac{|\tilde{f}(\mathbf{x}) - f(\mathbf{x})|}{\max_\mathbf{x} |f(\mathbf{x})|}$$

Cross-section	$\mathbf{e}^{(\mathbf{inf})}_{\mathbf{Tucker}}$	$\mathbf{e}^{(\mathbf{inf})}_{\mathbf{multilinear}}$	Remark
Σ_1	**31.76*10^{-5}**	**52.96*10^{-5}**	***The Tucker decomposition is more accurate than the multilinear interpolation.***
Σ_2	**34.99*10^{-5}**	**135.15*10^{-5}**	
Σ_3	**28.40*10^{-5}**	**45.613* 10^{-5}**	
Σ_4	**139.68*10^{-5}**	**1119.38*10^{-5}**	
Σ_5	**24.17*10^{-5}**	**196.98*10^{-5}**	
Σ_6	**21.58*10^{-5}**	**146.47*10^{-5}**	

Table 3: Comparison of the accuracy

Sparse representation (4) and (5):

- Reduced about of 50% of the storage size while keeping the same order of accuracy for the Tucker decomposition.

Conclusion and Perspectives

- **Main results:** in the comparison with the multilinear interpolation, the Tucker decomposition allows us to:
  - reduce the number of predetermined values ($\sim 38\%$).
  - reduce the storage size of predetermined data ($\sim 4 - 9$ times).
  - achieve a *better accuracy* for reconstructed functions.
- **Perspectives:**
  - The numerical method used to solve the integral equation in the Karhunen-Loève decomposition should be optimized in order to improve the [accuracy]/[complexity] ratio.
  - The criteria for *a priori* sparse representation needs to be investigated more in order to reduce the number of predetermined values.

Bibliography

References

[1] W. Hackbusch, Tensor spaces and numerical tensor calculus, Vol. 42, Springer Science & Business Media, 2012.

[2] L. De Lathauwer, B. De Moor, J. Vandewalle, A multilinear singular value decomposition, SIAM journal on Matrix Analysis and Applications 21 (4) (2000) 1253–1278.

[3] T. H. Luu, Y. Maday, M. Guillo, P. Guérin, A new method for reconstruction of cross-sections using Tucker decomposition, Journal of Computational Physics (in submission).

[4] Y. Maday, N. C. Nguyen, A. T. Patera, G. S. Pau, A general, multipurpose interpolation procedure: the magic points.

[5] T. H. Luu, Y. Maday, M. Guillo, P. Guérin, Reconstruction of neutronic macroscopic cross-sections using tucker decomposition, International Conference on Mathematics and Computation, Supercomputing in Nuclear Applications and the Monte Carlo Method (M&C 2015), Nashville, USA.