# Use cases of Tucker decomposition method for reconstruction of neutron macroscopic cross-sections

Thi Hieu Luu, Matthieu Guillo, Yvon Maday, Pierre Guérin

# Use cases of Tucker decomposition method for reconstruction of neutron macroscopic cross-sections

Thi Hieu LUU[a,b,*], Matthieu GUILLO[a], Yvon MADAY[b,c], Pierre GUÉRIN[a]

[a] *Département SINETICS, EDF Lab Paris-Saclay*
*7, Boulevard Gaspard Monge, 91120 Palaiseau, France*
[b] *Sorbonne Universités, UPMC Univ Paris 06, UMR 7598,*
*Laboratoire Jacques-Louis Lions, F-75005, Paris, France*
[c] *Institut Universitaire de France and*
*Division of Applied Mathematics, Brown University, Providence, RI, USA*

**Abstract**

The neutron cross-sections are inputs for nuclear reactor core simulations, they depend on various physical parameters. Because of industrial constraint (e.g. calculation time), the cross-sections can not be calculated on the fly due to the huge number of them. Hence, a reconstruction (or interpolation) process is used in order to evaluate the cross-sections at every point required, from (as few as possible) pre-calculated points. With most classical methods (for example: multilinear interpolation which is used in the core code COCAGNE of EDF (Électricité De France)), high accuracy for the reconstruction often requires a lot of pre-calculated points. We propose to use the Tucker decomposition, a low-rank tensor approximation method, to deal with this problem. The Tucker decomposition allows us to capture the most important information (one parameter at a time) to reconstruct the cross-sections. This information is stored as basis functions (called *tensor directional basis functions*) and the *coefficients* of the decomposition instead of pre-calculated points. Full reconstruction is done at the core code level using these decompositions. In this paper, a simplified multivariate analysis technique (based on statistical analysis) is also proposed in order to demonstrate that we can improve the quality of the acquired information as well as the accuracy of our approach. Using the Tucker decomposition, we will show in proposed use cases that we can reduce significantly the number of pre-calculated points and the storage size (compared to the multilinear interpolations) while achieving high accuracy for the reconstruction, even on a larger domain of parameters.

*Keywords:* cross-section reconstruction, Tucker decomposition, interpolation, low-rank tensor approximation, statistical analysis

*Corresponding author

*Email addresses:* luu@ljll.math.upmc.fr (Thi Hieu LUU), matthieu.guillo@edf.fr (Matthieu GUILLO), maday@ann.jussieu.fr (Yvon MADAY), pierre.guerin@edf.fr (Pierre GUÉRIN)

## 1. Introduction

Like most companies working in nuclear electricity production, Électricité De France in its research and development (EDF R&D) departments develops highly accurate nuclear reactor core simulator system. Two main classes of approach are employed for simulations : deterministic and probabilistic. Our work relates to the deterministic one.

The purpose of a core simulator system is to be able to simulate any kind of physical quantity for the proper operation and the safety of the power plant. In order to do that, one has to solve Boltzmann's equations (or an approximation of these) for neutrons. These equations need, at every (physical-)cell of the 3D space, some physical inputs, the cross-sections, denoted by the letter $\Sigma$, that model the interactions between fission-induced neutrons and nuclei from either the fuel or the moderator (water in the case of PWR).

These cross-sections vary from one (physical-)point of the core to another - hence $\Sigma = \Sigma(\overrightarrow{P})$ with $\overrightarrow{P} = (x, y, z)$ - and depend on $d$ local parameters (so called feedback parameters), such as: *burnup (bu), fuel temperature ($t_f$), moderator density ($\rho_m$), boron concentration ($b_c$), xenon level (xe), etc.* Each feedback parameter can be seen as one axis of a $d$-dimensional space called *parameter-phase space* and a set of parameter values as a point in this parameter-phase space. Therefore, for each cell in a 3D space, there is a $d$-tuple parameter's value in the $d$-dimensional parameter-phase space. The actual value of $d$ is model dependent.

In order to do that, core simulator systems classically involve two solvers used in chain : a lattice code and a core code. At the EDF in the department SINETICS (SImulation NEutronique, Technologie de l'Information, Calcul Scientifique), we use, in a first step, the lattice code APOLLO2 [1], [2] developed at CEA, that generates cross-sections in the parameter-phase space and store them inside a file. This file acts like a database; we call it a " *nuclear library*". Then, in a second step, the core code COCAGNE [3] reads this library and, for every cell in the 3D space, computes the values of the $d$ parameters (using for instance thermal-hydraulic code, depletion loop and so on) and evaluates the values of the cross-sections using the database and a reconstruction model. The reconstruction allows from values in the nuclear library (the smaller in size the better) to get an approximate value at any point in the parameter-phase space (the more accurate the better).

For the simulation of nuclear reactor core, one generally defines a particular point in the parameter-phase space at which the reactor core operates normally. This point is called *nominal point/condition*. Actually, the *burnup* direction is not involved in the definition of the nominal point since it is related to time. The domain near the nominal point (which thus lives in $\mathbb{R}^{d-1}$) is referred to as the *standard domain*. It is composed of values of the parameters in the parameter-phase space that are encountered under standard working conditions of the power plant. On the contrary, it is called the *extended domain* when the parameters get out of these running situations and are encountered in some special operations for the reactor or incidental situations. There is no much

precise definition of these domains that may represent different objects through various publications.

In our proposed use cases, the standard domain and the extended domains are hypercube in $\mathbb{R}^5$. Here, the standard domain is a subset of extended domains as is precisely presented in subsection 5.3 and approximately presented in subsection 5.4.

In general, high accuracy for the reconstruction requires a lot of pre-calculated points, i.e., a lot of lattice calculations. The total number of pre-calculations performed by the lattice code is usually so large ($\sim$ thousands) that it takes a lot of calculation time (while a calculation takes about 30 seconds). This becomes more cumbersome for an extended domain with a model such as multilinear interpolation. The reduction of the number of these calculations as much as possible has been the motivation for introducing in [4] a new reconstruction based on a low-rank tensor approximation method, that is referred to as *the Tucker decomposition*. The purpose of the current paper is to present and test on various assemblies (UOX, UOX-Gd, MOX, on standard and extended domains) this new reconstruction and compare it to the currently used multilinear interpolator in COCAGNE.

In this paper, we also propose to use some new techniques, such as: the Empirical Interpolation Method (EIM) and the pre/post-analysis in order to achieve better accuracy for our method, compared to results presented in the paper [5]. Moreover, results of new use cases (performed on extended domains) will be shown.

The paper is organized in the following manner:

- Section 2 gives an overview of different methods for the reconstruction of cross-sections used by different utility companies.
- In section 3, we present the Tucker decomposition applied to a set of multivariate functions.
- Section 4 illustrates how we can efficiently use the Tucker decomposition for the cross-section reconstruction problem and how we can deduce cross-section properties in high dimension. In this section, we will show the advantages of our model, which allow us to pre-analyze and post-analyze our approach.
- Section 5 is reserved to our proposed use cases, applied to different fuel assemblies (UOX, UOX-Gd, MOX). The results obtained demonstrate that we can reduce the pre-calculated data while achieving high accuracy in both cases: the standard domain and the extended domain. We also present some problems that we encountered in this section.
- Section 6 is dedicated to conclusion and perspective.

## 2. Overview of different methods for the reconstruction of neutron cross-sections

There are many core simulator systems for nuclear reactor simulations, for example: the pair of softwares ARCADIA(HERMES)-ARTEMIS [6] used by AREVA, DRAGON-DONJON [7] used by Canadian Nuclear Society, NEXUS-ANC [8] used by Westinghouse, etc. Each such system employs a model in order to reconstruct the cross-sections. These models can be classified in two main categories:

- The cross-section values are approximated by adding some perturbation or correction terms to values calculated at or around the nominal point, see e.g. [9], [10], [8], [11]. In general, the correction terms are based on physical knowledge or some expansion techniques, such as the Taylor expansion.

- The cross-section values are interpolated from the pre-calculated values on a grid (for instance, Cartesian grid constructed with a tensor product of the discretized points on each axis [12] , sparse grid [13], quasi-random grid [14], etc.). The interpolation techniques in these models are different by the choice of basis functions: piece-wise linear functions, B-spline, Lagrange polynomials, etc. COCAGNE belongs to this category where the Cartesian grid and the piece-wise linear functions are used in the multi-linear interpolation model.

The first category's methods have been proven suitable for the simulation on a "standard" domain where the parameter-values are rather close to the nominal condition. Of course, when they are far away from this condition, their accuracy is lost since the heuristics used around nominal values may not be valid anymore. The second ones are more general but high accuracy requires a lot of pre-calculated data. For instance, the multilinear approach that is used at EDF, relies on the acquisition of the values of the various cross sections on a Cartesian grid of the (standard or extended) domain in the parameter-phase space, i.e. in $\mathbb{R}^5$ from the lattice code. This nuclear library has a cardinal equal to $N^d$, where $N$ stands for the number of discretized points in each phase direction. Then the reconstruction allows us to build an approximation of each cross section at any point by: i) locating this point to one of the cells of the Cartesian grid (to which this point belongs), here each cell is seen as a $d$ dimensional object and its vertices are the points on which the value of each cross section is available, ii) averaging the previous values in a convex way to provide a linear approximation in each dimension. This is a very simple methods, and its accuracy (second oder in $L^2$ or $L^\infty$ norms scaling like $\mathcal{O}(N^{-2})$) in terms of size of the cell that implies to have a quite large library on standard domains and very - even too much - large nuclear library on extended domains.

Therefore, either we have a high efficiency reconstruction method (meaning high accuracy with few points) but on a specific domain only, or we have an expensive reconstruction method (meaning high accuracy at a lot of pre-calculated points) but suitable for any domain.

We present in the next section a new approach in this field [4], based on the Tucker decomposition [15], [16], [5].

## 3. Reconstruction model based on the Tucker decomposition

Let $f(\mathbf{x}) = f(x_1, \ldots, x_d)$ be a multivariate (say continuous) function defined on $\mathbf{\Omega} = \Omega_1 \times \ldots \times \Omega_d \subset \mathbb{R}^d$. Let, for any direction $j$ in $\mathbb{R}^d$, be a given basis $\{\varphi_{i_j}^{(j)}\}_{i_j=1}^{\infty}$ say in $L^2(\Omega_j)$, then $\{\prod_{j=1}^{d} \varphi_{i_j}^{(j)}\}_{i_j=1}^{\infty}$ is a basis of $L^2(\mathbf{\Omega})$ and there exist coefficients $\mathbf{a}_{i_1 \ldots i_d}$ such that:

$$f(\mathbf{x}) = \sum_{i_1=1}^{\infty} \cdots \sum_{i_d=1}^{\infty} \underbrace{\mathbf{a}_{i_1 \ldots i_d}}_{\text{coefficient}} \prod_{j=1}^{d} \underbrace{\varphi_{i_j}^{(j)}(x_j)}_{\text{tensor directional basis function}} \tag{1}$$

In order to be interesting for our purpose, we would like that a truncated approximation

$$f(\mathbf{x}) \approx \tilde{f}(x_1, \ldots, x_d) = \sum_{i_1=1}^{r_1} \cdots \sum_{i_d=1}^{r_d} \underbrace{\mathbf{a}_{i_1 \ldots i_d}}_{\text{coefficient}} \prod_{j=1}^{d} \underbrace{\varphi_{i_j}^{(j)}(x_j)}_{\text{tensor directional basis function}} \tag{2}$$

be accurate enough with very few terms providing a low-rank tensor approximation of $f$, i.e., an approximation by a small combination of tensor products of one-variate functions.

This is precisely the frame for Tucker decomposition where the focus is on determining:

1. The proper basis set, called "*tensor directional basis functions*" $\{\varphi_{i_j}^{(j)}\}_{i_j=1}^{r_j}$ for each direction $j$, $1 \leq j \leq d$.
2. The total of $R = \prod_{j=1}^{d} r_j$ coefficients $\mathbf{a}_i = \mathbf{a}_{i_1 \ldots i_d}$

In our previous article [4], we have proposed to define the tensor directional basis functions by an extension of the Karhunen-Loève decomposition, known as Higher-Order Singular Value Decomposition (HOSVD) technique (see [17] and section 8.3, page 230 of the book [16]). From the property of the Karhunen-Loève decomposition, this allows us to provide a condensed approximation, where the number $r_j$ of tensor directional basis functions used in each direction is small. Note that, in order to get a proper condensed approximation, these tensor directional basis functions are not chosen a priori, but deeply depend on $f$ in order to achieve the second item above, they are thus not well suited to approximate another multivariate function than $f$. We recall the construction of the tensor directional basis functions in the next subsection.

Next, the coefficients are obtained by solving a system of linear equations. This system requires inputs as the values of the function $f$ on a set of points selected by using the Empirical Interpolation Method (EIM) (see [18], [19]).

Let us recall briefly the methodology.

*3.1. Determination of tensor directional basis functions*

The tensor directional basis functions are constructed, direction by direction, by extracting the important information (principal component) in the studied direction. The technique used to extract information is based on the Karhunen-Loève decomposition and the HOSVD technique.

We present briefly here the Karhunen-Loève decomposition (KL). This decomposition is also known as: Principal Component Analysis (PCA) , Proper Orthogonal Decomposition (POD), which depends on the science community. With this decomposition, we can decompose a two-variate function $f(x,y) \in L^2(\Omega_x \times \Omega_y)$ in a infinite separate terms:

$$f(x,y) = \sum_{n=1}^{\infty} \sqrt{\lambda_n}\varphi_n(x)\psi_n(y)$$

where $(\lambda_n, \varphi_n)$ is the $n^{\text{th}}$ couple of solution: eigenvalue - eigenfunction of the problem:

Finding $(\lambda, \varphi) \in \mathbb{R} \times L^2(\Omega_x)$ of the Hilbert-Schmidt operator $K_f^{(x)}$, such that:

$$\boxed{K_f^{(x)}\varphi \equiv \int_{\Omega_x}\int_{\Omega_y} f(x,y)f(x',y)\varphi(x')dx'dy = \lambda\varphi(x),\ \forall x \in \Omega_x} \tag{3}$$

The couple $(\lambda_n, \psi_n)$ is the $n^{\text{th}}$ solution of a similar operator $K_f^{(y)}$. $K_f^{(x)}$ and $K_f^{(y)}$ have the same eigenvalues $\lambda$ and they are all positive by the theory of Hilbert-Schmidt operators. If we rank these eigenvalues in the decreasing order, they decrease quickly in general: $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k \gg \ldots \gg \lambda_n \geq \ldots > 0$. Moreover, for a given $r \in \mathbb{N}$, if we search to approximate a function $f(x,y) \in L^2(\Omega_x \times \Omega_y)$ by a combination of $r$ separate variable functions:

$$f(x,y) \approx \tilde{f}_r = \sum_{n=1}^{r} a_n u_n(x) v_n(y),\ \forall (x,y) \in \Omega_x \times \Omega_y$$

then the Karhunen-Loève decomposition provides the best approximation $\tilde{f}_r^{best}$ in the sense that minimizes the root mean square error (RMSE):

$$e^{RMSE} = \sqrt{||f(x,y) - \tilde{f}_r(x,y)||^2_{L^2(\Omega)}} = \sqrt{\int_{\Omega_x}\int_{\Omega_y} (f(x,y) - \tilde{f}_r(x,y))^2 dxdy}$$

Here,

$$\tilde{f}_r^{best} = f_{KL_{|r}} = \sum_{n=1}^{r} \sqrt{\lambda_n}\varphi_n(x)\psi_n(y); \qquad e_r^{RMSE,\,best} = \sqrt{\sum_{n=r+1}^{\infty} \lambda_n}$$

The Karhunen-Loève decomposition has no direct extension for $d$-variate functions, where $d \geq 3$. Therefore, we propose to use the condensation technique

in the HOSVD in order to exploit the Karhunen-Loève decomposition for the construction of tensor directional basis functions in our Tucker decomposition.

In our case, we consider $f = f(x_1, \ldots, x_d)$ as a $d$-variate function defined on $\mathbf{\Omega} = \Omega_1 \times \ldots \times \Omega_d \subset \mathbb{R}^d$ and $f \in L^2(\mathbf{\Omega})$. Under such condition, $K_f^{(x)}$ (or $K_f^{(y)}$) is an operator bounded in $L^2$. We apply now the idea described above to the Tucker decomposition of $f$.

Concretely, the tensor directional basis functions for each direction $j$ ($1 \leq j \leq d$) are chosen among the eigenfunctions of a directional Hilbert-Schmidt operator $K_f^{(j)}$, where:

$$\boxed{K_f^{(j)}\varphi^{(j)}(x_j) \equiv \int_{\Omega_j}\int_{\Omega_{\mathbf{y}}} f(x_j, \mathbf{y}) f(x'_j, \mathbf{y})\varphi^{(j)}(x'_j)dx'_j d\mathbf{y} = \lambda\varphi^{(j)}(x_j),\ \forall x_j \in \Omega_j} \quad (4)$$

Here, $K_f^{(j)}$ corresponds to the $K_f^{(x)}$ in the Karhunen-Loève decomposition with $x := x_j \in \Omega_j$ and $\mathbf{y} := (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d) \in \Omega_{\mathbf{y}} := \Omega_1 \times \ldots \times \Omega_{j-1} \times \Omega_{j+1} \times \ldots \times \Omega_d$.

We thus keep the first $r_j$ eigenfunctions - being the tensor directional basis functions in the $j$-th direction, and the eigenvalues - indicating the degree of approximation.

In practice, we keep only the eigenfunctions associated with the eigenvalues $\lambda_i$ satisfying: $\dfrac{\lambda_i}{\lambda_1} > \epsilon$, with $\lambda_1 \geq \lambda_2 \geq \ldots > 0$. Here, $\epsilon$ is a parameter that can be chosen by the user.

Problem (4) has to be solved numerically by discretizing the domain $\Omega_j \times \Omega_{\mathbf{y}}$ of $f$ into $\Omega_{N_j} \times \Omega_{N_{\mathbf{y}}}$, where $\#\Omega_{N_j} = N_j$, $\#\Omega_{N_{\mathbf{y}}} = N_{\mathbf{y}}$. The values of $f$ on the discretized points $(x_p, \mathbf{y}_q)$ are denoted by:

$$f_{pq} = f(x_p, \mathbf{y}_q),\ \text{with}\ (x_p, \mathbf{y}_q) \in \Omega_{N_j} \times \Omega_{N_{\mathbf{y}}}. \quad (5)$$

The integrals appearing in the Hilbert-Schmidt operator are approximated by numerical integration. The integral weight at $(x_p, \mathbf{y}_q)$ is denoted by $\Delta_{pq}$. When such a discretization is performed, equation (4) cannot be satisfied $\forall x_j \in \Omega_j$. The discrete version of problem (4) can be written as the following discrete equations:

$$\sum_{p=0}^{N_j}\sum_{q=0}^{N_y} f_{kq} f_{pq} \Delta_{pq} (\overrightarrow{\varphi}^{(j)})_p = \lambda (\overrightarrow{\varphi}^{(j)})_k \quad (6)$$

On the left hand side of the equations (6), the values $f_{pq}$ of $f$ at discretized points are required. In our case where $f$ are cross-sections $\Sigma$, these values are computed by lattice code APOLLO2.

In order to reduce the number of required values $f_{pq}$ while capturing information efficiently for each direction, we have proposed to consider the following numerical integration based on:

- A fine discretization in the direction $j$ with $N_j$ quadrature points.

- A coarse discretization in the other directions k ($k \neq j$) with only 2 points, providing $2^{d-1}$ points.

Such a discretization corresponds to a grid, referred to as $x_j$*-grid* in order to mention which parameter is concerned by this grid.

In order to get high accuracy approximation with the quadrature rules for the integral equations (4), we chose the Clenshaw-Curtis points [20], [21] for the fine discretizations.

The procedure to construct tensor directional basis functions per direction results in having as many $x_j$-grids as the number of parameters. This procedure is illustrated by figure 1.

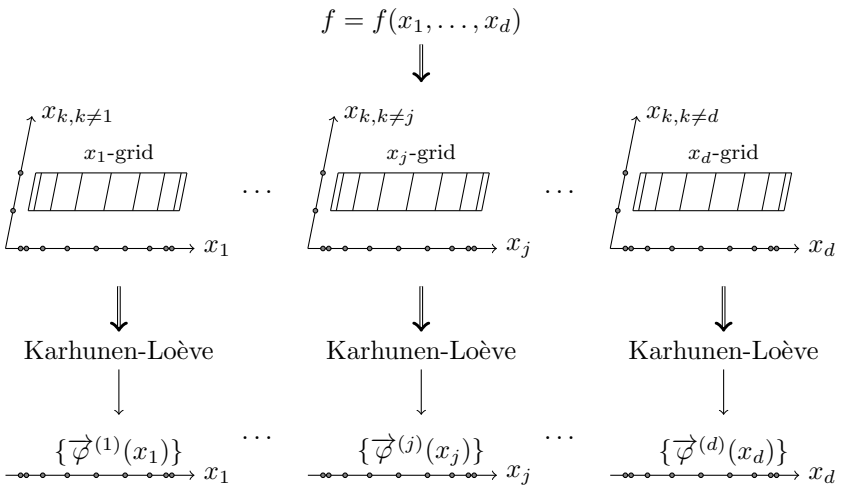



Figure 1: Construction of the tensor directional basis functions using the $x_j$-grids and the Karhunen-Loève decomposition, direction by direction.

After the discretization of the problem (4) what is obtained by solving the linear system (6), for each direction $j$, the vectors $\{\overrightarrow{\varphi}_\ell^{(j)}\}_\ell$ are thus naturally interpreted as approximated values for the plain eigenfunctions in (4).

$$\overrightarrow{\varphi}^{(j)} = (\varphi^{(j)}(x_p))_{x_p \in \Omega_{N_j}} \in \mathbb{R}^{N_j}$$

Note that we use the Clenshaw-Curtis points for the numerical rule is in connection with a polynomial approximation of the functions in the direction $j$. Hence, in order to reconstruct the tensor directional basis function from these values, we choose the Lagrange interpolation (see illustration in figure 2).

### *3.2. Determination of coefficients*

After determining the tensor directional basis functions in the Tucker decomposition (2), we explain how to determine the $R = \prod_{j=1}^{d} r_j$ associated coefficients $\mathbf{a}_i$, where $r_j$ is the number of the tensor directional basis functions in

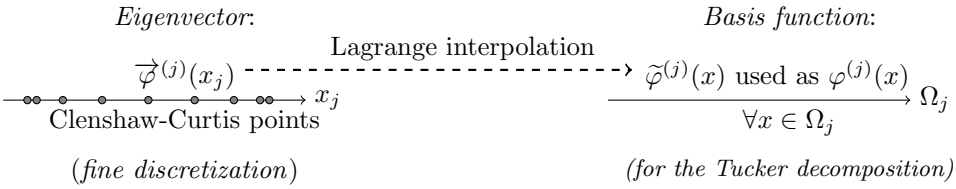



Figure 2: Lagrange interpolation in order to reconstruct the tensor directional basis functions from the eigenvectors for a direction $j$.

the direction $j$. We propose to choose these coefficients as the solution of the following system:

$$\begin{cases} \sum_{i=1}^{R} \mathbf{a}_i \prod_{j=1}^{d} \varphi_{i_j}^{(j)}(x_{1_j}) = f(\mathbf{x}_1) \\ \dots \\ \sum_{i=1}^{R} \mathbf{a}_i \prod_{j=1}^{d} \varphi_{i_j}^{(j)}(x_{t_j}) = f(\mathbf{x}_t) \\ \dots \\ \sum_{i=1}^{R} \mathbf{a}_i \prod_{j=1}^{d} \varphi_{i_j}^{(j)}(x_{R_j}) = f(\mathbf{x}_R) \end{cases} \tag{7}$$

where $\mathbf{x}_t = (x_{t_1}, \dots, x_{t_j}, \dots, x_{t_d}) \in \Omega \subset \mathbb{R}^d$.

It means that the approximation by the Tucker decomposition is exact at $R$ points of the set $\{\mathbf{x}_t\}_{t=1}^{R}$ (the Tucker decomposition becomes an interpolation of $f$ on these $R$ points).

At this stage, new values of the function $f$ (values of cross-sections performed by the lattice code APOLLO2 in our application) are required on the right hand side of the system (7). Indeed, the points: $\mathbf{x}_1, \dots, \mathbf{x}_R$ do not belong to any of the previous $x_j$-grids because they are constructed as a tensor product of points extracted from the fine discretizations of the $x_j$-grids (see later).

The choice of the set $\{\mathbf{x}_t\}_{t=1}^{R}$ is not trivial because we can take any point $\mathbf{x} \in \Omega$ to constitute such a set. But the accuracy of the Tucker decomposition depends on the choice of the set $\{\mathbf{x}_t\}_{t=1}^{R}$ when (7) is imposed. To deal with this problem, we propose to constitute the set $\{\mathbf{x}_t\}_{t=1}^{R}$ as a tensor product of 1D-sets $\{x_{t_j}^{(j)}\}_{t_j=1}^{r_j}$:

$$\{\mathbf{x}_t\}_{t=1}^{R} = \times_{j=1}^{d} \{x_{t_j}^{(j)}\}_{t_j=1}^{r_j} \tag{8}$$

where $\{x_{t_j}^{(j)}\}_{t_j=1}^{r_j}$ are selected among the $N_j$ ($N_j > r_j$) points of the fine Clenshaw-Curtis discretization of the direction $j$. For each axis, the EIM [18] is applied to the set of tensor directional basis functions $\{\varphi_{i_j}^{(j)}\}_{i_j=1}^{r_j}$ and the corresponding fine discretization in the direction $j$ in order to find the $r_j$ points of the set $\{x_{t_j}^{(j)}\}_{t_j=1}^{r_j}$.

We present generally and briefly the EIM used in our application. This is an iterative method, each time we enrich the set $\mathcal{X}$ of interpolation points and improve the quality of interpolation for a set of functions $\mathcal{G}$:

- Let $\omega \subset \mathbb{R}^n$ ($1 \leq n \in \mathbb{N}$), $\mathcal{G} = \{\varphi : \omega \to \mathbb{R} | \, \varphi \in L^\infty(\omega)\}$, $\#\mathcal{G} = K$ , $P \in \mathbb{N}$.

- For $p = 1$ :
  $u_1 = argmax_{\varphi \in \mathcal{G}} ||\varphi||_{L^\infty(\omega)}$,
  $x_1 = argmax_{x \in \omega} |u_1(x)|$
  - Set $\mathcal{X}_1 = \{x_1\}$
  - Establish the interpolation operator $\mathcal{I}_1$:

$$\forall \varphi \in \mathcal{G} : \mathcal{I}_1[\varphi] = \frac{\varphi(x_1)}{u_1(x_1)} u_1(.)$$

- For $p > 1$ :
  $u_{p+1} = argmax_{\varphi \in \mathcal{G}} ||\varphi - \mathcal{I}_p[\varphi]||_{L^\infty(\omega)}$,
  $x_{p+1} = argmax_{x \in \omega} |u_{p+1}(x) - \mathcal{I}_p[u_{p+1}](x)|$
  - Set $\mathcal{X}_{p+1} = \mathcal{X}_p \bigcup \{x_{p+1}\}$
  - Establish the interpolation operator $\mathcal{I}_{p+1}$, that satisfies:

    $\forall \varphi \in \mathcal{G} : \mathcal{I}_{p+1}[\varphi] =$ Lagrange interpolation via the $p+1$ points of $\mathcal{X}_{p+1}$

    i.e.:

$$\mathcal{I}_{p+1}[\varphi] = \mathcal{I}_p[\varphi] + \frac{(\varphi - \mathcal{I}_p(\varphi))(x_{p+1})}{(u_{p+1} - \mathcal{I}_p(u_{p+1}))(x_{p+1})} (u_{p+1} - \mathcal{I}_p(u_{p+1}))$$

- Continue until the imposed value, i.e. $p = P$.

In our work, we are interested in the construction of the set $\mathcal{X}_P$ with a given $P$ but we do not care about the interpolation operators $\mathcal{I}_p$.

If we have to reconstruct not only one function $f$ but a set of functions $\{f_k\}_k$ (as in our application we need to reconstruct many cross-sections), the set of points $\{\mathbf{x}_t\}_{t=1}^R$ needs to be constituted such that it is suitable for the use of all functions $f_k$. In this case, the EIM is recursively employed for the determination of the set $\{x_{t_j}^{(j)}\}_{t_j=1}^{r_j}$ on each axis (see [4] for more details).

At this stage and in our application, we need to compute the values of all cross-sections $\{f_k\}_k$ by APOLLO2 code on a new grid. The values obtained will be used on the right hand side of each system (7). This new grid is referred to as *the Tucker grid* which is related to the coefficients $\mathbf{a}$ of the Tucker decomposition. Therefore, in our work, we use a total of $d+1$ grids where the $d$ first grids are used to determine the tensor directional basis functions and the last one is used to determine the coefficients of the Tucker decomposition.

## 4. Application of the Tucker decomposition to the reconstruction of cross-sections

### *4.1. Cross-section notions*

In neutron physics, the cross-sections are denoted by $\Sigma_r^g$ where $r$ designates the reaction kind and $g$ designates the energy group. For our applications to the COCAGNE software, we refer to some cross-sections like the

`macro totale` - $\Sigma_t^g$, the `macro absorption` - $\Sigma_a^g$, `macro fission` - $\Sigma_f^g$ and the `macro nu*fission` - $\nu\Sigma_f^g$. A particular case, the `macro scattering` also depends on anisotropy order $o$, departure energy group $g$ and arrival energy group $g^{'}$, denoted by $\Sigma_{so}^{g\to g^{'}}$.

In our work, there are two energy groups considered: fast group ($g = 1$) and thermal group ($g = 2$). The cross-sections depend on 5 parameters: *burnup, boron concentration, moderator density, fuel temperature* and *xenon level*. Here *xenon level* is the ratio of the xenon concentration computed at the core level (using a xenon depletion model chosen by the user) and the xenon concentration computed by APOLLO2 at the assemble level and stored in the library. Therefore, in the Tucker decomposition, we have 5 grids in the determination of the tensor directional basis functions, each one corresponds to a direction (parameter) and 1 grid corresponding to the resolution of the coefficients.

### *4.2. Reference grid*

In order to measure the accuracy of our reconstruction method, we use a Cartesian grid, referred to as *reference grid*. This grid is very fine (with around 10,000 points for a standard domain and with around 14,000 points for an extended domain). The discretized values on each axis of the reference grid are chosen such that they are different from the fine discretization of the 5 first grids and relatively randomized. The cross-section values performed by the APOLLO2 code [1], [2] on the reference grid are *exact values* ($\Sigma_r^g$).

The values of the cross-sections evaluated by the Tucker decomposition on the reference grid are *approximated values* ($\widetilde{\Sigma}_r^g$).

From the exact and approximated values on the reference grid, we can evaluate the error of our approach and test the quality of the reconstruction. We illustrate the use of the reference grid in the diagram presented in figure 3.

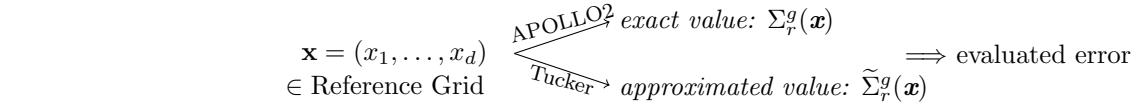



Figure 3: Reference grid in order to measure the accuracy of our approach.

In order to compute the relative error for cross-sections according to a formula similar to: $\dfrac{\Sigma_r^g - \tilde{\Sigma}_r^g}{\Sigma_r^g}$, we remark that this error is very small, percent is not appropriate. As is classical in the neutron community, we chose to use *pcm* (0.001 %) instead that is more suited for this kind of study.

As we saw in figure 3, each point $\mathbf{x}$ of the reference grid can be associated with an evaluated error. In our work and for cross-sections, this evaluated error

is a relative one which is defined by:

$$e_{\Sigma_r^g,\,\text{Relative}}(\mathbf{x}) = \underbrace{\frac{\Sigma_r^g(\mathbf{x}) - \tilde{\Sigma}_r^g(\mathbf{x})}{\max_{\mathbf{x}} |\Sigma_r^g(\mathbf{x})|} * 10^5}_{\text{Relative Error (pcm)}}, \;\; \mathbf{x} \in \textit{reference grid}$$

The "max" is employed in the denominator of the relative error in order to correctly deal with small cross-sections over the region of the reference grid. This normalization allows us to compare the accuracy between reconstructed cross-sections.

In order to have an estimate for the accuracy of our approach, a final error is defined as follows:

$$e_{\Sigma_r^g}(pcm) = \max_{\mathbf{x} \in \textit{reference grid}} |e_{\Sigma_r^g,\,\text{Relative}}(\mathbf{x})| \tag{9}$$

On the other hand, we want to get a better understanding of our approximation with the small cross-sections in the region of the reference grid, we therefore propose another measure based on root mean square (RMS) of local relative errors:

$$e_{\Sigma_r^g}^{RMS}(pcm) = \sqrt{\frac{\sum_{i=1}^{N} [\frac{\Sigma_r^g(\mathbf{x}_i) - \tilde{\Sigma}_r^g(\mathbf{x}_i)}{\Sigma_r^g(\mathbf{x}_i)}]^2}{N}} * 10^5 \tag{10}$$

where $\mathbf{x}_i \in$ *reference grid*, $N = \#(\textit{reference grid})$.

### *4.3. Analyses used in order to achieve high accuracy for the reconstruction problem*

As we saw, the Tucker decomposition depends on the chosen tensor directional basis functions and these bases are constructed axis by axis from the eigenvectors of the extended Karhunen-Loève decomposition. Therefore, information represented in the tensor directional basis functions as well as the number of eigenvectors taken for each axis impact directly the accuracy of the Tucker method. These two characters are exploited in our analyses (pre-analysis and post-analysis) in order to propose adaptive solutions to achieve the [desired accuracy]/[storage] ratio for the reconstruction problem.

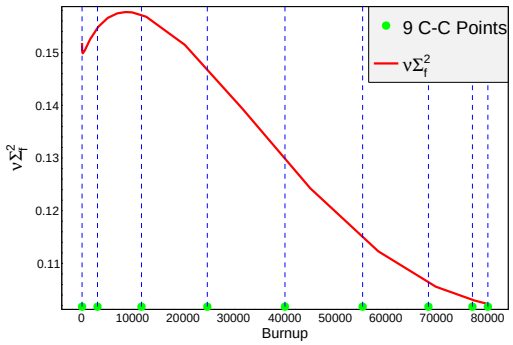
#### *4.3.1. Pre-analysis of cross-sections as functions of each parameter*

In our work, as we explained in the previous section, the approximation of the functions is done by polynomials, this approximation is global and of high order, this is true at least if the regularity is coherent over the whole domain $\Omega_j$. Therefore, if the cross-sections vary much in some zones, the approximation must be adjusted in order to capture the variation, either by increasing the degree of polynomials or by splitting up the domain $\Omega_j$ in sub-domains (like in the spectral element method). At some points however, increasing the degree

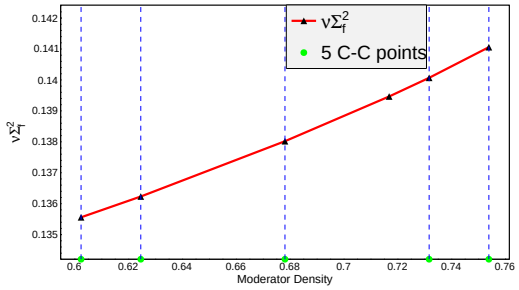
may lead to instabilities, hence we limit the order of the Lagrange polynomials in our approach to 8, thus using a maximum of 9 Clenshaw-Curtis points.

In order to analyze cross-section variation and adjust our approach, the idea is thus to analyze each cross-section as a function of one parameter at a time, which allows us to see *a priori* the variation of cross-sections.

In practice, we study the variation of cross-sections for each parameter by varying only this parameter while the others are fixed at nominal values. We show in figure 4 an example with the cross-section $\nu\Sigma_f^2$, where the UOX assembly is studied on the standard domain.




(a) $\nu\Sigma_f^2$ as a function of burnup. The discretization with 9 Clenshaw-Curtis (C-C) points (represented in small circles) is not enough at low *burnup* values.




(b) $\nu\Sigma_f^2$ as a function of moderator density. Clenshaw-Curtis (C-C) points (represented in small circles) are suitable in this case. (*The triangle symbols represent the values of* $\nu\Sigma_f^2$ *at the Clenshaw-Curtis points, in which, the nominal point* ($\sim 0.72$) *is automatically added by the simulator system*).

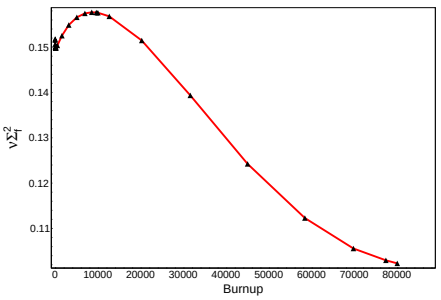
Figure 4: Variation of the cross-section $\nu\Sigma_f^2$ as a function of one parameter (case of the UOX assembly on the standard domain).

In the figure 4, we see that the curve of $\nu\Sigma_f^2$ as a function of *burnup* varies a lot (see figure 4a) while for the other parameters, it is quite linear, e.g., figure

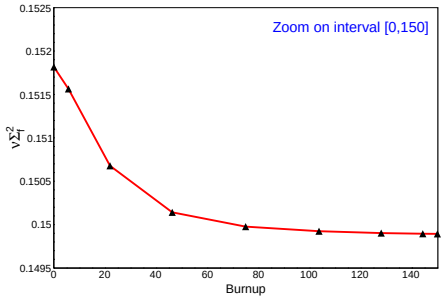
4b for the parameter *moderator density*. Hence, we need more discretized points on the *burnup* axis than on any other axes.

If we use a polynomial approximation of degree 8, hence using 9 Clenshaw-Curtis points on the *burnup* axis (these points are represented by small circles on the *burnup* axis in figure 4a), the first two discretized values obtained are: $0.0\,(MWd/t)$ and $3044.81\,(MWd/t)$. We can see that this discretization can not capture the information about cross-section variations due to strong variations at the low *burnup* values, that is caused by the *xenon* effect before its concentration reaches equilibrium. Since the low *burnup* is around the interval $[0, 150]$ while $3044.81 \gg 150$, we therefore propose to sub-divide the whole interval $[0, 80000]$ into 3 sub-intervals: $[0, 150]$, $[150, 10000]$ and $[10000, 80000]$. For the case studied here (UOX assembly on the standard domain), we choose 9 Clenshaw-Curtis points for each sub-interval (there are thus two common points: 150 and 10000). The new discretization with 25 points in total is now well suited to the variation of the cross-section $\nu\Sigma_f^2$, as presented in figure 5.

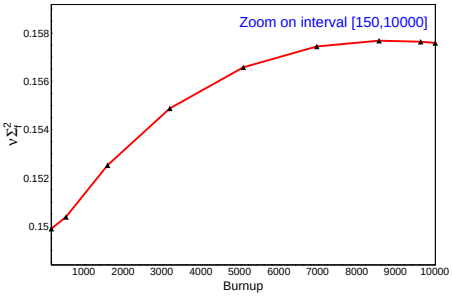
This technique, interval subdivision, can be used each time when the fine discretization can not capture cross-section variations.



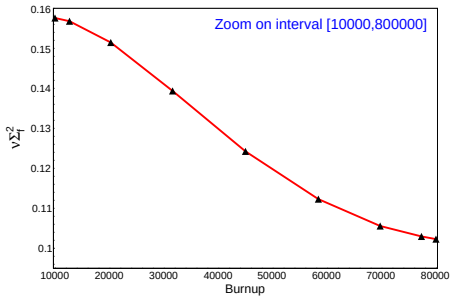

(a) 25 Clenshaw-Curtis points on [0,80000].




(b) 9 Clenshaw-Curtis points on [0,150].




(c) 9 Clenshaw-Curtis points on [150,10000].




(d) 9 Clenshaw-Curtis points on [10000,80000].

Figure 5: Subdivision of the *burnup* axis (with 9 Clenshaw-Curtis points on each sub-interval) in order to capture the cross-section variation (case of the UOX assembly on the standard domain).

*4.3.2. Post-analysis of evaluated errors*

*Histogram of evaluated errors.* In our work, each point $\mathbf{x}$ of the reference grid (in the parameter-phase space) has 5 coordinates: $x_1, \ldots, x_5$. In practice, we store successively these coordinates and the evaluated errors associated with the point $\mathbf{x}$ into files. This allows us to present and analyze the evaluated errors as a function of each parameter.

From such stored files, we present the evaluated errors as *histograms*. A histogram contains some bins used to divide the entire evaluated errors into adjacent intervals. The height of a bin represents the number of values (evaluated errors in our case) that fall into this bin. In our test, we want to present percentages in order to maintain the scale (in %) when we change the number of entries; we thus normalize the bins by dividing the height of each bin by the total number of entries (the number of points in the reference grid). The representation of evaluated errors by histogram allows us to see how the evaluated errors distribute and which the error zones need to be analyzed.

In general, the centered histogram with small standard deviations is an ideal result, which means that most of the evaluated errors are around 0. Note however that an analysis of the deviated distributions can be exploited in order to determine where they come from and which parameters are responsible for this behavior. These analyses allow us to improve the accuracy and error distributions as we will show in the following section.

*Post-analysis via histogram of evaluated errors.* The accuracy of the Tucker decomposition depends on the number of eigenvectors (or tensor directional basis functions) taken for each direction. It also depends on the fine discretization realized on each axis because tensor directional basis functions are constructed from these discretizations. Improving one of these two factors may improve the accuracy of our approach. Therefore, they are the solutions to the problems revealed by our analyses, as we will show here.

In order to illustrate our analysis techniques, we present an example in the case of the UOX-Gd assembly on the extended domain with the cross-section $\Sigma_t^1$ (fast group). The distribution of relative errors is shown in figure 6. This distribution varies in the interval [-55 pcm, 35 pcm] and looks like a Gaussian distribution with a trailing part ($\sim [-55\,pcm, -15\,pcm]$).

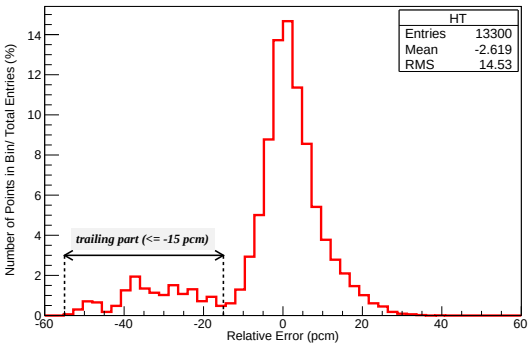
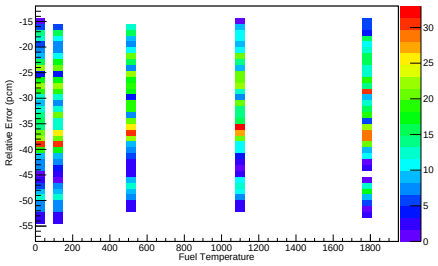


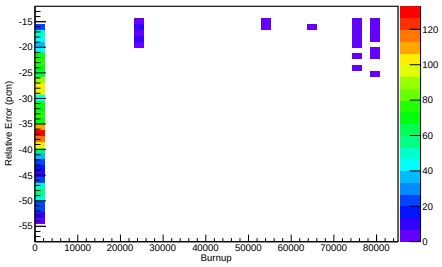

Figure 6: Relative errors of $\Sigma_t^1$ (deviation distribution) before being improved.

In order to improve the accuracy as well as the distribution of relative errors of the cross-section $\Sigma_t^1$, we will analyze the trailing part ($-55\,pcm \leq$ error $\leq -15\,pcm$) to see if its behavior depends on a particular parameter. Analysis per parameter is one of methods which helps us to see how the errors depend on each parameter and which specific values are involved.

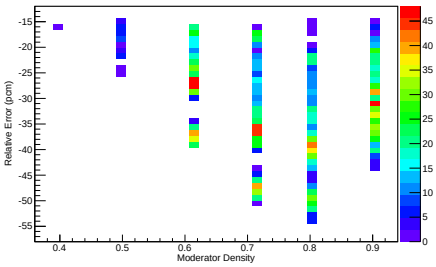
Applying this idea, we obtained the distribution of the trailing part as a function of respectively each parameter: *burnup, boron concentration, moderator density, fuel temperature* and *xenon level*. In figure 7, these error distributions are presented as the 2D-histograms, in the following manner:




(a) Trailing part ($\leq -15\,pcm$) as a function of *fuel temperature*.




(b) Trailing part ($\leq -15\,pcm$) as a function of *burnup*.




(c) Trailing part ($\leq -15\,pcm$) as a function of *moderator density*.

Figure 7: Trailing part ($\leq -15\,pcm$) of $\Sigma_t^1$ (fast group) as a function of one parameter.

- The error distributions are represented by the variations of colors in each *column inside* a figure. These columns are placed above each discretized value (used for the fine discretization of the analyzed parameter) to describe how an error value depends on this fixed discretized value while varying the values of other parameters.
- Each color corresponds to a number of points occurred (or density of points) as described in the column on the right of each figure.

From the error distributions shown in figure 7, we can see that:

- For each of the following parameters: *fuel temperature, boron concentration* and *xenon level*, its discretized values are all presented, see figure 7a for an example about the parameter *fuel temperature*. In this figure, the error distributions are presented by 5 columns corresponding to all 5 discretized values of the parameter *fuel temperature*. We can see that these 5 columns are similar to each other in length and density distribution of occurred points (i.e. color pattern). Therefore, we can say that the error distributions are quite homogeneous and do not depend on a particular discretized value of the *fuel temperature* parameter. (The same conclusion for *boron concentration* and *xenon level* parameter from their corresponding distributions not shown here).

- For each of the following parameters: *burnup* and *moderator density*, the error distributions are quite different and vary as a function of only some particular values, for example, around 0 for the case of *burnup* (figure 7b), and around 0.8 for the case of *moderator density* (figure 7c).

Using this analysis, we can conclude that *burnup* and *moderator density* are major factors which are the cause of the trailing part. Hence, we propose to add more tensor directional basis functions on these two directions. Concretely, we can improve the accuracy with 7 tensor directional basis functions for burnup (instead of 4) and 4 tensor directional basis functions for moderator density (instead of 3). The new result is shown in figure 8.

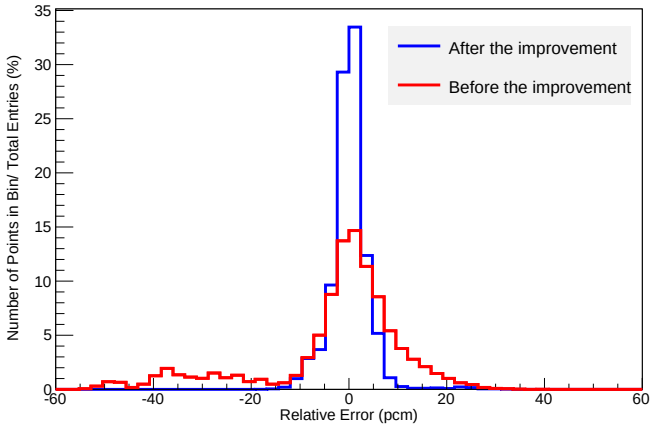



Figure 8: Relative errors of $\Sigma_t^1$ before (in red) and after (in blue) being improved (number of tensor directional basis functions for *burnup* axis and *moderator density* axis are increased).

We see that the relative errors are improved, they vary in the interval [-15 pcm, 30 pcm] and become more centered. For this improvement, we did not increase the number of discretized points on the $x_j$-grids in the direction of *burnup* nor *moderator density* axis, but we added more tensor directional basis

functions for these axes. The reason we did not try to increase discretized points on these axes is that this technique requires more APOLLO2 calculations. If we want to improve the accuracy by changing the fine discretization, then the zones of the *burnup* around 0 and the *moderator density* around 0.8 need to be better discretized.

We summarize here the main steps in the post-analysis of evaluated errors:

- Finding the major parameters which are the cause of analyzed error zones:
  - Analyzing the distribution of evaluated errors as a function of each parameter.
  - Determining the parameters and the particular discretized values on which the accuracy does not satisfy our requirement.
- Once the major parameters are identified, accuracy can be improved by either of the following methods:
  - Adding more tensor directional basis functions in these directions. This can require new APOLLO2 calculations if the points used in the system for coefficients (7) are changed due to the new tensor directional basis functions.
  - Changing the axial discretization in order to take into account the parameter values that are responsible for large error. The new APOLLO2 calculations will be required on new $x_j$-grids which corresponds to new discretizations and on a new Tucker grid which is used in the systems (7) for the coefficients of the Tucker decomposition .

In general, these improvements need more APOLLO2 calculations but the first solution (adding more tensor directional basis functions) often requires less calculations than the second one.

### *4.4. Cost of the Tucker decomposition compared to the multilinear interpolation*

#### *4.4.1. Number of APOLLO2 calculations*

In the Tucker decomposition, the APOLLO2 calculations are performed in two stages:

- First, the APOLLO2 calculations are used for the $d$ $x_j$-grids in order to solve numerically the integral equation (4) which provides the tensor directional basis functions.
- Then, the APOLLO2 calculations are used for the cross-section values on the right hand side of the system (7) in order to determine the coefficients of the Tucker decomposition.

Therefore, the number of APOLLO2 calculations is:

$$\underbrace{\sum_{j=1}^{d} N_j * 2^{d-1}}_{\text{for d } x_j\text{-grids}} \quad + \quad \underbrace{\prod_{j=1}^{d} r_j^{max}}_{\text{for points on the right hand side of (7)}} \tag{11}$$

where $N_j$ is the number of points of the fine discretization in direction $j$ and $r_j^{max}$ is the maximal number of tensor directional basis functions in direction $j$, over all cross-sections.

In the multilinear interpolation, we compute cross-section values on a Cartesian grid. Therefore, if each axis has $N_j$ points, the number of APOLLO2 calculations for the multilinear interpolation is:

$$\prod_{j=1}^{d} N_j \tag{12}$$

*4.4.2. Storage size in the neutron libraries*

In the Tucker decomposition, the stored data for one cross-section includes eigenvector values for all directions ($r_j$ eigenvectors for a direction $j$) and $R = \prod_{j=1}^{d} r_j$ coefficients. Therefore, the storage size is cross-section dependent. Note that $r_j$ depends on the criterion used to select the number of tensor directional basis functions (see the section 3.1) and the decreasing speed to zero of eigenvalues in the problems (4), i.e. depending on the complexity of each considered cross-section. The number of stored floating points for each cross-section is:

$$\underbrace{\sum_{j=1}^{d} r_j * N_j}_{\text{for tensor directional basis functions}} + \underbrace{\prod_{j=1}^{d} r_j}_{\text{for coefficients}} \tag{13}$$

On the contrary, the storage sizes for the multilinear interpolation are the same for all cross-sections because we store all cross-section values performed on the Cartesian grid. Therefore, the number of stored floating points for the multilinear interpolation is:

$$\prod_{j=1}^{d} N_j \tag{14}$$

*4.4.3. Number of floating point operations for the reconstruction*

We recall here the Tucker decomposition formula:

$$f(\mathbf{x}) \approx \tilde{f}(x_1, \ldots, x_d) = \sum_{i_1=1}^{r_1} \cdots \sum_{i_d=1}^{r_d} \underbrace{\mathbf{a}_{i_1 \ldots i_d}}_{\text{coefficient}} \prod_{j=1}^{d} \underbrace{\varphi_{i_j}^{(j)}(x_j)}_{\text{tensor directional basis function}}$$

where in our application, each $\varphi_{i_j}^{(j)}$ is a $N_j$ degree polynomial (If $N_j + 1$ is the number of points in the fine discretization of $x_j$-grids). Hence, in order to calculate each term $\mathbf{a}_{i_1 \ldots i_d} \prod_{j=1}^{d} \varphi_{i_j}^{(j)}(x_j)$, we need the following number of floating point operations:

$$\underbrace{(\sum_{j=1}^{d} N_j + d + 1)}_{multiplications} + \underbrace{\sum_{j=1}^{d} N_j}_{additions}$$

The total number of floating point operations for the Tucker decomposition is equal to:

$$\prod_{j=1}^{d} r_j [\underbrace{(\sum_{j=1}^{d} N_j + d + 1)}_{multiplications} + \underbrace{\sum_{j=1}^{d} N_j}_{additions} ]$$

If $r_j = r$ and $N_j = N$, $1 \leq j \leq d$, the number of floating point operations is of the order of $\mathcal{O}(dr^d N)$. In the case of the multilinear interpolation, this number is of the order of $\mathcal{O}(2^{d+1})$.

*4.4.4. Summary*

We summarize the comparison of the two methods in the table 1.

| | Multilinear Interpolation | Tucker Decomposition |
|---|---|---|
| Number of APOLLO2 calculations | $\mathcal{O}(N^d)$ | $\mathcal{O}(dN2^{d-1} + r^d)$ |
| Storage | $\mathcal{O}(N^d)$ (*the same for all cross-sections*) | $\mathcal{O}(dNr + r^d)$ (*depending on each cross-section*) |
| Number of floating point operations | $\mathcal{O}(2^{d+1})$ (*the same for all cross-sections*) (*time-consuming process for finding $2^d$ neighbors of interpolation point*) | $\mathcal{O}(dNr^d)$ (*depending on each cross-section*) (*can be reduced by sparse technique (see perspective)*) |
| Remark | $r \leqslant N$, $r \sim 5$ | |

Table 1: Comparison of the two methods: multilinear interpolation and Tucker decomposition

## 5. Proposed use cases

*5.1. Reactivity*

About the reconstruction of cross-sections, the following points must be noted:

- Cross-sections are not the ones used in the core code because they need to be corrected in order to take into account leakage, equivalence, historical correction, ...before entering the flux solver.
- Cross-sections are merely inputs for the flux solver.

We are therefore interested in outputs of the flux solver, such as: $k_{eff}$ or reactivity. For an infinite medium, the neutron flux and the cross-sections are constant on the parameter-phase space. Therefore, with this simplified hypothesis and with two energy groups, we do not need a flux solver, $k_{eff}$ becomes now $k_\infty$ and the following analytic formula are applied (see page 1172, 1173, 1221 of the book [22]):

$$\text{reactivity} = 1 - \frac{1}{k_{eff}}, \text{with } k_{eff} = k_\infty = \frac{\nu\Sigma_f^1 * (\Sigma_t^2 - \Sigma_{s0}^{2\to 2}) + \nu\Sigma_f^2 * \Sigma_{s0}^{1\to 2}}{(\Sigma_t^1 - \Sigma_{s0}^{1\to 1}) * (\Sigma_t^2 - \Sigma_{s0}^{2\to 2}) - \Sigma_{s0}^{1\to 2} * \Sigma_{s0}^{2\to 1}} \quad (15)$$

A final indicator for the accuracy of the reactivity is expressed as follows:

$$\mathbf{e}_{\text{Reactivity}}(pcm) = \max_{\mathbf{x} \in reference\ grid} |e_{\text{Reactivity, Absolute}}(\mathbf{x})| \tag{16}$$

where:

$$e_{\text{Reactivity, Absolute}}(\mathbf{x}) = \underbrace{(\frac{1}{k_\infty(\mathbf{x})} - \frac{1}{\tilde{k}_\infty(\mathbf{x})}) * 10^5}_{\text{Absolute Error (pcm)}} \tag{17}$$

As for the measure of reconstructed cross-section errors, we also propose another indicator based on the root mean square of relative errors:

$$e^{RMS}_{\text{Reactivity}}(pcm) = \sqrt{\frac{\sum_{i=1}^{N} [\frac{(1 - \frac{1}{k_\infty(\mathbf{x}_i)}) - (1 - \frac{1}{\tilde{k}_\infty(\mathbf{x}_i)})}{1 - \frac{1}{k_\infty(\mathbf{x}_i)}}]^2}{N}} * 10^5 \tag{18}$$

where $\mathbf{x}_i \in$ *reference grid*, $N = \#($*reference grid*$)$.

### *5.2. Grids used in use cases*

In the following use cases, the cross-sections depend on 5 parameters: *burnup, boron concentration, moderator density, fuel temperature* and *xenon level*. Therefore, we have 5 $x_j$-grids in the determination of tensor directional basis functions, each one corresponds to one direction in the Tucker decomposition.

For each use case, we use a reference grid (see section 4.2) in order to measure the accuracy of the Tucker decomposition.

### *5.3. Use case on the standard domain with UOX assembly*

In this case, we consider an UOX assembly, 3.7% enrichment for a 900MWe-PWR (Pressurized Water Reactor). The cross-sections depend on 5 parameters which vary in the intervals given in table 2.

| Parameter name | min value | max value |
|---|---|---|
| Burnup, $MWd/t$ | 0.0 | 80000.0 |
| Fuel temperature,$^{\circ}C$ | 286.0 | 1100.0 |
| Moderator density, $g/cm^3$ | 0.602 | 0.753 |
| Boron concentration, $ppm$ | 0.0 | 1800.0 |
| Xenon level, % | 0.0 | 1.0 |

Table 2: Parameters and their intervals.

In the 5 $x_j$-grids, only the grid for the *burnup* axis is sub-divided into 3 intervals: [0,150], [150,10000] and [10000, 80000]. Each sub-interval has 9 Clenshaw-Curtis points, we therefore have 25 points in this direction (two common points

for three intervals). The 4 other grids have 5 Clenshaw-Curtis points for the fine discretization without using subdivision. With these 5 grids, we have a total of $25 * 2^4 + 4 * (5 * 2^4) = 720$ points. The reference grid described in section 4.2 contains 9,300 points.

The accuracy (in pcm) of the Tucker decomposition is shown in table 3. We see that the relative errors of cross-sections are smaller than 100 pcm in general. These accuracies are better than ones of the multilinear interpolation employed in COCAGNE with the currently used number of points (shown in table 4).

| Cross Section | $\mathbf{e_{Tucker}}$ (pcm) formula (9) for cross-sections formula (16) for reactivity | $\mathbf{e}^{RMS}_{\mathbf{Tucker}}$ (pcm) formula (10) for cross-sections formula (18) for reactivity |
|---|---|---|
| $\Sigma_t^1$ | 28 | 0.07 |
| $\Sigma_t^2$ | 40 | 0.09 |
| $\Sigma_a^1$ | 38 | 0.11 |
| $\Sigma_a^2$ | 49 | 0.25 |
| $\nu\Sigma_f^1$ | 61 | 0.24 |
| $\nu\Sigma_f^2$ | 79 | 0.40 |
| $\Sigma_{s0}^{1\to1}$ | 27 | 0.07 |
| $\Sigma_{s0}^{1\to2}$ | 19 | 0.06 |
| $\Sigma_{s0}^{2\to1}$ | 100 | 0.56 |
| $\Sigma_{s0}^{2\to2}$ | 15 | 0.04 |
| $\Sigma_f^1$ | 66 | 0.26 |
| $\Sigma_f^2$ | 74 | 0.41 |
| **Reactivity** | 387 | 0.9 |

Table 3: Accuracy of the Tucker decomposition over 9,300 points of the reference grid for the UOX assembly on the standard domain.

We observed that the accuracy of the cross-section $\Sigma_{s0}^{2\to1}$ is the worst result obtained by the Tucker decomposition. It is also the worst case for the multilinear interpolation. As we will see, these observations are valid for all use cases presented in this paper.

In order to achieve the accuracy in the table 3, we employed the number of APOLLO2 calculations and the storage size respectively presented in table 4 and table 5. They are compared to those of the multilinear interpolation.

| **Number of APOLLO2 calculations** | | | | |
|---|---|---|---|---|
| Tucker decomposition | | | | Multilinear Interpolation |
| $\underbrace{720}_{\text{for the 5 } x_j\text{-grids}}$ | + | $\underbrace{378}_{\text{for the Tucker grids related to the systems (7)}}$ | =**1098** | **1782** |

Table 4: Comparison of the number of APOLLO2 calculations between the Tucker decomposition and the multilinear interpolation for the UOX assembly on the standard domain.

We note that the number of APOLLO2 calculations (1782) in the table 4 for the multilinear interpolation comes from an optimized industrial process in COCAGNE whereas with the Tucker decomposition, this is not so much the case since this is a preliminary work. Even though, these results show that the

| Cross section | **Storage (number of floats)** | |
|---|---|---|
| | Tucker | Multilinear |
| $\Sigma_t^1$ | 204 | 1782 |
| $\Sigma_t^2$ | 192 | idem |
| $\Sigma_a^1$ | 416 | idem |
| $\Sigma_a^2$ | 355 | idem |
| $\nu\Sigma_f^1$ | 388 | idem |
| $\nu\Sigma_f^2$ | 290 | idem |
| $\Sigma_{s0}^{1\to1}$ | 204 | idem |
| $\Sigma_{s0}^{1\to2}$ | 529 | idem |
| $\Sigma_{s0}^{2\to1}$ | 371 | idem |
| $\Sigma_{s0}^{2\to2}$ | 192 | idem |
| $\Sigma_f^1$ | 388 | idem |
| $\Sigma_f^2$ | 290 | idem |

Table 5: Comparison of the storage for each cross-section, between the Tucker decomposition and the multilinear interpolation for the UOX assembly on the standard domain.

Tucker decomposition reduced by 38% of the number of APOLLO2 calculations and by a factor of 3.4 to 8.7 of the storage size (depending on the studied cross-section).

### *5.4. Use cases for the extended domain*

The extended domain is tricky for many reconstruction models, especially those involved with corrections from values computed around the nominal values (Taylor expansions, heuristics, etc.). The multilinear interpolation does not suffer from this problem. However, to have high quality cross-sections, it requires a lot of points.

These following use cases are performed on an extended domain where the values of four parameters: *boron concentration (*$\sim [0\,ppm, 2200\,ppm]$*), moderator density (*$\sim [0.3\,g/cm^3, 1\,g/cm^3]$*), fuel temperature (*$\sim [10℃, 2000℃]$*)* and *xenon level (*$\sim [0, 3]$*)* are larger than those of the standard domain (the interval of *burnup* does not change). Again, precise values issued from the industrial scheme can not be shown here.

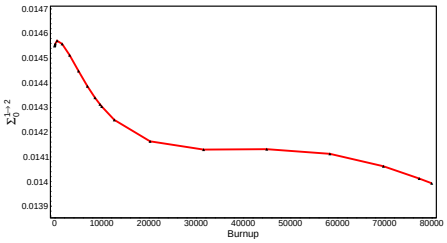
For all the following test cases on the extended domain, the Tucker decomposition employees the same 5 $x_j$-grids. By chance, the number of total points on these 5 grids is the same as in the standard domain case (720 points) but the discretizations are completely different. We summarize here *the fine discretization* (using the Clenshaw-Curtis points) for each $x_j$-grid:

- *Burnup* grid: sub-divided into 3 intervals: [0,150] with 3 points, [150,10000] with 9 points and [10000, 80000] with 9 points.
- *Moderator density* grid: sub-divided into 2 intervals, each sub-interval has 5 points.
- *Fuel temperature* grid: 5 points.

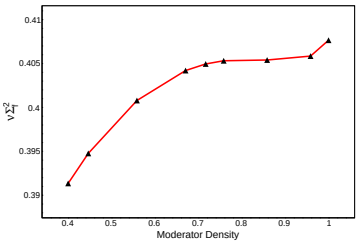
- *Boron concentration* grid: 7 points.
- *Xenon level*: 5 points.

Therefore, in our use cases on the extended domain, we have a total of $19 * 2^4 + 9 * 2^4 + 5 * 2^4 + 7 * 2^4 + 5 * 2^4 = 720$ points for the 5 $x_j$-grids.

In these 5 $x_j$-grids, subdivisions are applied to both the *burnup* axis and the *moderator density* axis. This comes from the method discussed in section 4.3.1 in order to capture the variation of cross-sections. The great variations (for the UOX and the MOX assemblies) as a function of *burnup* (presented in figure 9a) or *moderator density* (presented in figure 9b) explain why subdivisions on these two axes are needed.




(a) Variation of $\Sigma_{s0}^{1\to 2}$ as a function of *burnup* (case of UOX assembly on extended domain).




(b) Variation of $\nu\Sigma_f^2$ as a function of *moderator density* (case of MOX assembly on extended domain).

Figure 9: Variations of some cross-sections lead to subdivisions on the extended domain for the *burnup* axis and the *moderator density* axis.

*5.4.1. UOX assembly on the extended domain*

Once again, the UOX assembly, 3.7% enrichment is considered but on the extended domain. In this case, the reference grid contains 14,700 points.

Evaluated errors for some cross-sections and for the reactivity is shown in table 6. We see that the relative errors are about 100 pcm, except for $\Sigma_{s0}^{2\to 1}$ (464 pcm). These accuracies are better than those of the multilinear method for all cases shown in the table 6.

The comparisons of the Tucker decomposition with the multilinear interpolation on the number of APOLLO2 calculations and on the storage size are respectively shown in table 7 and table 8. The number of APOLLO2 calculations (3060) used for the multilinear interpolation is already optimized in the code COCAGNE. We see that the number of APOLLO2 calculations is reduced by 14.7% and the storage size is reduced by a factor of 1.65 to 11.16 by using the Tucker decomposition.

| Cross Section | $\mathbf{e_{Tucker}}$ (pcm)<br>formula (9) for cross-sections<br>formula (16) for reactivity | $\mathbf{e}^{RMS}_{\mathbf{Tucker}}$ (pcm)<br>formula (10) for cross-sections<br>formula (18) for reactivity |
|---|---|---|
| $\Sigma_t^1$ | 9 | 0.01 |
| $\Sigma_t^2$ | 86 | 0.36 |
| $\Sigma_a^1$ | 64 | 0.82 |
| $\Sigma_a^2$ | 97 | 0.39 |
| $\nu\Sigma_f^1$ | 47 | 0.05 |
| $\nu\Sigma_f^2$ | 104 | 0.38 |
| $\Sigma_{s0}^{1\to1}$ | 8 | 0.01 |
| $\Sigma_{s0}^{1\to2}$ | 23 | 0.06 |
| $\Sigma_{s0}^{2\to1}$ | 464 | 2.6 |
| $\Sigma_{s0}^{2\to2}$ | 84 | 0.35 |
| $\Sigma_f^1$ | 43 | 0.07 |
| $\Sigma_f^2$ | 102 | 0.38 |
| **Reactivity** | 179 | 0.37 |

Table 6: Accuracy of the Tucker decomposition over 14,700 points of the reference grid for the UOX assembly on the extended domain.

| **Number of APOLLO2 calculations** | | | |
|---|---|---|---|
| Tucker decomposition | | | Multilinear Interpolation |
| $\underbrace{720}_{\text{for the 5 } x_j\text{-grids}}$ | + | $\underbrace{1890}_{\text{for the Tucker grids related to the systems (7)}}$ | =**2610** | **3060** |

Table 7: Comparison of the number of APOLLO2 calculations between the Tucker decomposition and the multilinear interpolation for the UOX assembly on the extended domain.

| Cross section | **Storage (number of floats)** | |
|---|---|---|
| | Tucker | Multilinear |
| $\Sigma_t^1$ | 343 | 3060 |
| $\Sigma_t^2$ | 343 | idem |
| $\Sigma_a^1$ | 527 | idem |
| $\Sigma_a^2$ | 866 | idem |
| $\nu\Sigma_f^1$ | 537 | idem |
| $\nu\Sigma_f^2$ | 636 | idem |
| $\Sigma_{s0}^{1\to1}$ | 343 | idem |
| $\Sigma_{s0}^{1\to2}$ | 854 | idem |
| $\Sigma_{s0}^{2\to1}$ | 1849 | idem |
| $\Sigma_{s0}^{2\to2}$ | 274 | idem |
| $\Sigma_f^1$ | 537 | idem |
| $\Sigma_f^2$ | 636 | idem |

Table 8: Comparison of the storage for each cross-section, between the Tucker decomposition and the multilinear interpolation for the UOX assembly on the extended domain.

*5.4.2. UOX-Gd assembly on the extended domain*

This use case studies an UOX-Gd assembly for a 1300MWe-PWR. The reference grid contains 13,000 points.

The accuracy of the Tucker decomposition is shown in table 9. In this test case, the relative errors of the cross-sections as well as the absolute errors of the reactivity are the worst compared to the other cases (UOX, MOX (see later)). But in these results, we did not perform any improvement (e.g., change the discretization, add more points on some axis, etc., as shown in the section 4.3.2) in order to get better results. We see in the table 9 that the relative errors are smaller than 200 pcm in general, except for $\Sigma_{s0}^{2\to1}$ (550 pcm) and $\Sigma_a^2$ (327 pcm). (However, these results are all better than accuracies obtained by the multilinear interpolation).

| Cross Section | $\mathbf{e_{Tucker}}$ (pcm)<br>formula (9) for cross-sections<br>formula (16) for reactivity | $\mathbf{e}_{\mathbf{Tucker}}^{RMS}$ (pcm)<br>formula (10) for cross-sections<br>formula (18) for reactivity |
|---|---|---|
| $\Sigma_t^1$ | 54 | 0.16 |
| $\Sigma_t^2$ | 93 | 0.36 |
| $\Sigma_a^1$ | 77 | 0.15 |
| $\Sigma_a^2$ | 327 | 0.64 |
| $\nu\Sigma_f^1$ | 76 | 0.18 |
| $\nu\Sigma_f^2$ | 175 | 0.45 |
| $\Sigma_{s0}^{1\to1}$ | 38 | 0.12 |
| $\Sigma_{s0}^{1\to2}$ | 150 | 0.32 |
| $\Sigma_{s0}^{2\to1}$ | 550 | 2.91 |
| $\Sigma_{s0}^{2\to2}$ | 87 | 0.36 |
| $\Sigma_f^1$ | 79 | 0.20 |
| $\Sigma_f^2$ | 175 | 0.46 |
| **Reactivity** | 535 | 1.30 |

Table 9: Accuracy of the Tucker decomposition over 13,000 points of the reference grid used for UOX-Gd assembly on the extended domain.

In order to maintain the order of accuracy, the multilinear model must take more points for its grid ($3060 \to 3960$ points), compared to previous use cases. Table 10 and 11 show that, by using the Tucker decomposition, the number of APOLLO2 calculations is reduced by 29.4% and the storage size is reduced by a factor of 3.6 to 20.84, compared to the multilinear interpolation.

| **Number of APOLLO2 calculations** | | | | |
|---|---|---|---|---|
| Tucker decomposition | | | | Multilinear Interpolation |
| $\underbrace{720}_{\text{for the 5 } x_j\text{-grids}}$ | + | $\underbrace{1440}_{\text{for the Tucker grids related to the systems (7)}}$ | **=2610** | **3960** |

Table 10: Comparison of the number of APOLLO2 calculations between the Tucker decomposition and the multilinear interpolation for UOX-Gd assembly on the extended domain.

| Cross section | **Storage (number of floats)** | |
|---|---|---|
| | Tucker | Multilinear |
| $\Sigma_t^1$ | 478 | 3960 |
| $\Sigma_t^2$ | 190 | idem |
| $\Sigma_a^1$ | 1070 | idem |
| $\Sigma_a^2$ | 537 | idem |
| $\nu\Sigma_f^1$ | 1098 | idem |
| $\nu\Sigma_f^2$ | 410 | idem |
| $\Sigma_{s0}^{1\to1}$ | 478 | idem |
| $\Sigma_{s0}^{1\to2}$ | 537 | idem |
| $\Sigma_{s0}^{2\to1}$ | 791 | idem |
| $\Sigma_{s0}^{2\to2}$ | 190 | idem |
| $\Sigma_f^1$ | 1098 | idem |
| $\Sigma_f^2$ | 410 | idem |

Table 11: Comparison of the storage for each cross-section, between the Tucker decomposition and the multilinear interpolation for UOX-Gd assembly on the extended domain.

*5.4.3. MOX assembly on the extended domain*

In this case, the MOX assembly (a mixture of uranium enriched with 2.5% and plutonium) is considered. For this use case, we use a reference grid containing 14,000 points.

| Cross Section | $\mathbf{e_{Tucker}}$ (pcm) formula (9) for cross-sections formula (16) for reactivity | $\mathbf{e_{Tucker}}$ (pcm) formula (10) for cross-sections |
|---|---|---|
| $\Sigma_t^1$ | 15 | 0.04 |
| $\Sigma_t^2$ | 58 | 0.24 |
| $\Sigma_a^1$ | 20 | 0.06 |
| $\Sigma_a^2$ | 28 | 0.10 |
| $\nu\Sigma_f^1$ | 11 | 0.03 |
| $\nu\Sigma_f^2$ | 24 | 0.10 |
| $\Sigma_{s0}^{1\to1}$ | 12 | 0.03 |
| $\Sigma_{s0}^{1\to2}$ | 11 | 0.06 |
| $\Sigma_{s0}^{2\to1}$ | 482 | 2.61 |
| $\Sigma_{s0}^{2\to2}$ | 62 | 0.28 |
| $\Sigma_f^1$ | 11 | 0.03 |
| $\Sigma_f^2$ | 25 | 0.10 |
| **Reactivity** | 93 | 0.24 |

Table 12: Accuracy of the Tucker decomposition over 14,000 points of the reference grid for MOX assembly on the extended domain

We show in table 12 the evaluated errors of the Tucker decomposition for some cross-sections and for the reactivity. We see that the relative errors are smaller than 100 pcm, except for the worst case $\Sigma_{s0}^{2\to1}$ (482 pcm). The absolute errors of the reactivity are smaller than 93 pcm. This is the best approximation results that we have (on the extended domain), not only for the accuracy but

also for the number of APOLLO2 calculations and the storage size in the neutron library.

| **Number of APOLLO2 calculations** | |
|---|---|
| Tucker decomposition | Multilinear Interpolation |
| $\underbrace{720}_{\text{for the 5 } x_j\text{-grids}} + \underbrace{720}_{\text{for the Tucker grids related to the systems (7)}} = \mathbf{1440}$ | **3060** |

Table 13: Comparison of the number of APOLLO2 calculations between the Tucker decomposition and the multilinear interpolation for MOX assembly on the extended domain.

The number of APOLLO2 calculations in table 13 and the storage size in table 14 are compared to those of the multilinear interpolation. These results show that, by using the Tucker decomposition, the number of APOLLO2 calculations is reduced by 52.9% and the storage size is reduced by a factor of 3.3 to 17.2.

| Cross section | **Storage (number of floats)** | |
|---|---|---|
| | Tucker | Multilinear |
| $\Sigma_t^1$ | 178 | 3060 |
| $\Sigma_t^2$ | 223 | idem |
| $\Sigma_a^1$ | 446 | idem |
| $\Sigma_a^2$ | 446 | idem |
| $\nu\Sigma_f^1$ | 286 | idem |
| $\nu\Sigma_f^2$ | 343 | idem |
| $\Sigma_{s0}^{1\to1}$ | 178 | idem |
| $\Sigma_{s0}^{1\to2}$ | 446 | idem |
| $\Sigma_{s0}^{2\to1}$ | 920 | idem |
| $\Sigma_{s0}^{2\to2}$ | 223 | idem |
| $\Sigma_f^1$ | 286 | idem |
| $\Sigma_f^2$ | 343 | idem |

Table 14: Comparison of the storage for each cross-section, between the Tucker decomposition and the multilinear interpolation for MOX assembly on the extended domain.

### *5.5. Discussion*

With the results obtained in the proposed use cases, we showed that the Tucker decomposition allows us to reduce the pre-calculated data as well as the storage size (compared to the multilinear model) while achieving high accuracy.

The subdivision (described in section 4.3.1) in order to capture the variations of cross-sections is applied to some axes, e.g. *burnup* axis is subdivided into three sub-intervals. Hence, for the *burnup* axis, we have three Lagrange polynomials (one for each sub-interval) which lead to the global reconstructed cross-sections are not of class $C^1$. This problem can be solved if for each segment, we choose a higher degree polynomial and absorb the remaining degree of freedom with the continuity of the function and first derivative.

We can also improve the accuracy of the Tucker decomposition by taking more tensor directional basis functions for some axes (see the section 4.3.2 with the example of the cross-section $\Sigma_t^1$ where we added tensor directional basis functions more for the burnup axis and the moderator density axis). Unfortunately, since the tensor directional basis functions are only the approximations of the eigenfunctions issued from the Karhunen-Loève decomposition, this solution could be unsuccessful in some cases. Hence, the method to find automatically the optimal number of tensor directional basis functions needs to be investigated.

The study has been only performed on macroscopic cross-sections but the same methodology could have been done on microscopic cross-sections. In fact, Tucker decomposition is performed for a cross-section at a given energy group and the variables are only the feedback parameters: there is no energy group axis. For a given energy group, each cross-section is seen as a new function of feedback parameters. Therefore the variation of cross-section due to the energy spectrum is not dealt with Tucker decomposition but with the change of cross-section energy group.

## 6. Conclusion and perspective

The Tucker decomposition described in this paper allows us to efficiently reconstruct the macroscopic neutron cross-sections. It also allows us to analyze the results obtained as a function of each parameter in order to improve the accuracy.

Using the Tucker decomposition, we reduce the number of APOLLO2 calculations (from 15% to 50%), the storage size in the neutron libraries (from 1.5 times to 20 times), compared to the multilinear interpolation. While precalculated data are significantly reduced, we still achieve high accuracy for the reconstructed cross-sections: around 100 pcm in relative errors, in general.

With a simplified hypothesis about an infinite medium and the two-group energy theory, the reactivity is analytically calculated. We can therefore verify the reactivity accuracy without using the flux solver. In general, the maximal absolute error for the reactivity are about 100 pcm to 600 pcm (depending on the use case). It is worth noting that the accuracy obtained using the Tucker decomposition is the same for the standard and the extended domains. No deterioration have been observed when moving from the standard to the extended domain.

However, the Tucker decomposition has some limitations. The evaluation step performed at the core code level could be expensive in CPU time because the Lagrange interpolation is used to reconstruct the tensor directional basis functions from the eigenvectors. Moreover, the Tucker decomposition does not ensure the positivity of cross-sections. The linearity relation, e.g. $\Sigma_t^1 = \Sigma_a^1 + \Sigma_{s0}^{1\to 2} + \Sigma_{s0}^{1\to 1}$ is also not exactly preserved (except of course if we define the approximation of $\Sigma_t^1$ as being the sum of the three approximations). However at this point, and for practical use, this does not seem to be an issue since, even without this linear property, Tucker decomposition gives a cross-section more

accurate than the multilinear one. There may be numerical side effects, unseen at that time, that will have to be checked using test cases once the method has been fully implemented in COCAGNE. If that is the case, we may need to add some correction factors. But until then there is no reason to believe that the loss of the linear property may be a problem except for code debugging purposes.

In future, we plan to study criteria which allow us to eliminate (*a priori* and *a posteriori*) the less important coefficients $\mathbf{a}$ in the representation of the Tucker decomposition. It means that the accuracy of the Tucker decomposition could be of the same order while many coefficients could be eliminated. This idea is feasible because the Tucker decomposition is constructed from "sorted" tensor directional basis functions (via the order of eigenvalues, e.g. decreasing order). Hence, there *a priori* exist less important coefficients, e.g. coefficients associated with the product of the last tensor directional basis functions. Therefore, such an elimination could lead to a similar accuracy while the number of APOLLO2 calculations and the storage size in the libraries would be further reduced.

## References


[1] R. Sanchez, J. Mondot, A. Cossic, I. Zmijarevic, et al., APOLLO II: A user-oriented, portable, modular code for multigroup transport assembly calculations, Nuclear Science and Engineering 100 (3) (1988) 352–362.

[2] R. Sanchez, I. Zmijarevic, M. Coste-Delclaux, E. Masiello, S. Santandrea, E. Martinolli, L. Villate, N. Schwartz, N. Guler, APOLLO2 year 2010, Nuclear Engineering and Technology 42 (2010) 474 – 499.

[3] L. Plagne, A. Ponçot, Generic programming for deterministic neutron transport codes, in: Mathematics and Computation, Supercomputing, Reactor Physics and Nuclear and biological Applications, 2005.

[4] T. H. Luu, Y. Maday, M. Guillo, P. Guérin, A new method for reconstruction of cross-sections using Tucker decomposition, Journal of Computational Physics (submitted). `https://hal.archives-ouvertes.fr/hal-01485419`.

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

[6] G. Hobson, H.-W. Bolloni, K.-A. Breith, R. van Geemert, H. Haase, B. Hartmann, ARTEMISTM Core Simulator: Latest developments, in: International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2013 (SNA + MC 2013), 2013.

[7] Polytechnique Montréal, DRAGON code, `http://www.polymtl.ca/nucleaire/en/logiciels/index.php`, accessed: 2016-10-15 (2016).

[8] E. Müller, L. Mayhue, B. Zhang, Reactor physics methods development at Westinghouse, in: Proc. Int. Conf. Nuclear Energy for New Europe 2007, 2007.

[9] T. Fujita, T. Endo, A. Yamamoto, A macroscopic cross-section model for BWR pin-by-pin core analysis, Journal of Nuclear Science and Technology 51 (3) (2014) 282–304.

[10] R. B. Turski, E. E. Morris, T. A. Taiwo, J. E. Cahalan, Macroscopic cross section generation and application for coupled spatial kinetics and thermal hydraulics analysis with SAS-DIF3DK.

[11] M. Stålek, C. Demazière, Development and validation of a cross-section interface for PARCS, Annals of Nuclear Energy 35 (12) (2008) 2397–2409.

[12] J. K. Watson, K. N. Ivanov, Improved cross-section modeling methodology for coupled three-dimensional transient simulations, Annals of Nuclear Energy 29 (8) (2002) 937–966.

[13] B. Danniëll, M. B. Pavel, Polynomial interpolation of few-group neutron cross sections on sparse grids, Annals of Nuclear Energy 64 (February 2014) 156–168.

[14] J. Dufek, Building the nodal nuclear data dependences in a many-dimensional state-variable space, Annals of Nuclear Energy 38 (7) (2011) 1569–1577.

[15] L. Tucker, Some mathematical notes on three-mode factor analysis, Psychometrika 31(3) (1966) 279–311.

[16] W. Hackbusch, Tensor spaces and numerical tensor calculus, Springer, 2012.

[17] 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.

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

[19] Y. Maday, O. Mula, G. Turinici, et al., A priori convergence of the generalized empirical interpolation method., in: 10th international conference on Sampling Theory and Applications (SampTA 2013), 2013, pp. 168–171.

[20] C. W. Clenshaw, A. R. Curtis, A method for numerical integration on an automatic computer, Numerische Mathematik 2(1) (1960) 197–205.

[21] J. P. Boyd, Chebyshev and Fourier spectral methods, Courier Dover Publications, 2001.

[22] S. Marguet, La physique des réacteurs nucléaires, 2ème édition, Tec& Doc, Lavoisier, 2013.