Generalizing the Column-Row Matrix Decomposition to Multi-way Arrays

Cesar Caiafa*†
ccaiafa@brain.riken.jp
LABSP, RIKEN Brain Science Institute, Wako,
Saitama 351-0198, JAPAN.
Phone: +81 48 467 9765, Fax: +81 48 467 9694.

Andrzej Cichocki‡
cia@brain.riken.jp
LABSP, RIKEN Brain Science Institute, Wako,
Saitama 351-0198, JAPAN.
Phone: +81 48 467 9668, Fax: +81 48 467 9686.

Abstract

In this paper, we provide two generalizations of the CUR matrix decomposition $\mathbf{Y} = \mathbf{CUR}$ (also known as pseudo-skeleton approximation method [1]) to the case of N-way arrays (tensors). These generalizations, which we called Fiber Sampling Tensor Decomposition types 1 and 2 (FSTD1 and FSTD2), provide explicit formulas for the parameters of a rank-$(R, R, ..., R)$ Tucker representation (the core tensor of size $R \times R \times ... \times R$ and the matrix factors of sizes $I_n \times R$, $n = 1, 2, ...N$) based only on some selected entries of the original tensor. FSTD1 uses P^{N-1} ($P \geq R$) n-mode fibers of the original tensor while FSTD2 uses exactly R fibers in each mode as matrix factors, as suggested by the existence theorem provided in [2], with a core tensor defined in terms of the the entries of a subtensor of size $R \times R \times ... \times R$. For $N = 2$ our results are reduced to the already known CUR matrix decomposition where the core matrix is defined as the inverse of the intersection submatrix, i.e. $\mathbf{U} = \mathbf{W}^{-1}$. Additionally, we provide an adaptive type algorithm for the selection of proper fibers in the FSTD1 model which is useful for large scale applications. Several numerical results are presented showing the performance of our FSTD1 Adaptive Algorithm compared to two recently proposed approximation methods for 3-way tensors.

*Corresponding Author.
†On leave from Engineering Faculty, University of Buenos Aires, ARGENTINA.
‡Also from Warsow University of Technology and Systems Research Institute, PAN, POLAND.

Keywords: Large-scale problems, low-rank approximation, N-way array, pseudo-skeleton (CUR) matrix decomposition, Tucker tensor decomposition.

1 Introduction

It is known that, given a matrix $\mathbf{Y} \in \mathbb{R}^{I \times J}$ with $\text{rank}(\mathbf{Y}) = R$, one can perfectly reconstruct it by choosing only $P = R$ rows and columns determining a non singular intersection submatrix $\mathbf{W} \in \mathbb{R}^{P \times P}$ and by calculating the corresponding CUR decomposition, i.e.

$$\mathbf{Y} = \mathbf{CUR}, \text{ with } \mathbf{U} = \mathbf{W}^{-1}, \tag{1}$$

where matrices $\mathbf{C} \in \mathbb{R}^{I \times P}$ and $\mathbf{R} \in \mathbb{R}^{P \times J}$ are the selected rows and columns of $\mathbf{Y}$ respectively.

This decomposition, also known as "skeleton" decomposition [3], is a classic of matrix analysis and provides a low rank approximation $\mathbf{Y} \approx \mathbf{CUR}$ when the number of selected rows and columns is lower than the $\text{rank}(\mathbf{Y})$ (i.e. for $P < R$). The proper selection of rows and columns and the core matrix $\mathbf{U}$ was subject of study in the literature, for example, Goreinov et al. developed a theory of "pseudo-skeleton" decompositions and established theoretical error bounds for the case of choosing an intersection matrix of maximum volume [4, 5, 1].

Low rank decompositions are useful for dealing with large scale problems, where a huge size matrix $\mathbf{Y}$ is approximated by a low rank representation ($P << I, J$) avoiding the storage of the entire matrix $\mathbf{Y}$. However, the optimal low rank representation is given by the truncated SVD, i.e. by using only its P dominant singular vectors, but it is too expensive to compute and it requires to access to all the entries of matrix $\mathbf{Y}$. For this reason, CUR decomposition becomes particularly attractive for large scale problems when the selection of rows and columns, and the calculation of the core matrix $\mathbf{U}$, can be done based solely on a reduced set of entries.

On the other hand, given an $N-$way array $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, a low rank tensor approximation is defined in terms of a reduced sum of rank-1 tensors such as the case of CANDECOMP/PARAFAC (CP) and Tucker models [6, 7]. Since in the rank-R matrix case there exist an exact CUR representation (equation (1)) which requires to know only the entries of the selected rows and columns, it is reasonable to ask if there is an equivalent representation of a tensor based only on few entries of it.

In this paper we introduce two different generalizations to tensors of the CUR matrix decomposition which we called Fiber Sampling Tensor Decomposition types 1 and 2 (FSTD1 and FSTD2). Both generalizations provide explicit formulas for calculating the parameters of a Tucker representation (core tensor and matrix factors) based only on some selected entries of the original tensor. More specifically, given a rank-$(R, R, ..., R)$ Tucker tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, FSTD1 provides a rank-$(R, R, ..., R)$ Tucker representation where the core tensor and the matrix factors are explicitly calculated based on a set of P^{N-1} ($P \geq R$) n-mode fibers. On the other hand, FSTD2 also provides a rank-$(R, R, ..., R)$

Tucker representation where the factor matrices are composed of the selected R n-mode fibers and the core tensor is explicitly calculated by using the entries of a carefully selected intersection subtensor of size $R \times R \times ... \times R$[1]. Additionally, we developed an adaptive algorithm for the selection of the fibers for FSTD1, which in the case of a 3-way tensor, it requires to sample exactly P^2 fibers from data. We also show, through numerical simulations, that this algorithm is robust providing good approximations even when the original tensor has not an exact Tucker representation.

1.1 Related works

It is worth to mention that there has been an increased interest on CUR decompositions for data analysis as an alternative to Principal Component Analysis (PCA) and SVD, specially for cases with sparse datasets. On this regard, the idea of CUR decomposition is to provide a representation of data as a linear combination of a few "meaningful" components which are exact replicas of columns and rows of the original data [8, 9, 10, 11, 12]. Existing methods for this kind of applications assume that the original data matrix $\mathbf{Y}$ can be large but is fully available for the computation of the CUR decomposition. In this case, the optimal choice for the core matrix is $\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{Y}\mathbf{R}^{\dagger}$ which demands rather heavy calculation by involving all the entries of matrix $\mathbf{Y}$. In these methods, also the rule for selecting rows and columns is obtained from the whole matrix $\mathbf{Y}$ by choosing them with probabilities associated to their ℓ_2 norms [9, 10, 11] or proportional to a special measure (scores) of the "statistical leverage" based on the dominant right singular vectors of matrix $\mathbf{Y}$ [12].

In [13, 14] Mahoney, Maggioni and Drineas have introduced their TensorCUR algorithm which applies the CUR matrix approximation [9] to one of the unfolding matrix modes (the "distinguished" mode) providing an approximation based on few tube fibers and few frontal slices or "slabs". However, the algorithms proposed in these works still require to access to all the entries since the randomization used is based on the knowledge of the fiber norms, otherwise these norms have to be taken from somewhere.

The idea of reconstructing tensors from a subset of its entries, without needing to access to all the entries, have been recently proposed in [15, 2, 16]. The existence of the pseudo-skeleton type approximation for 3-way tensors that are close to low rank tensors was first formulated and proved by Oseledets, Savostianov and Tyrtyshnikov in [2], and later also investigated by Goreinov in [17]. In the present work, we provide explicit formulas for obtaining these kind of decompositions. We restrict our discussion to CUR models, and their extension to tensors, that use only partial information of the original large-scale array by keeping the computational cost as low as possible.

[1] Our results are easily extended to the case of rank-$(R_1, R_2, ..., R_N)$ Tucker tensors but, for clarity of our presentation, we restrict it to the case $R = R_1 = R_2 = ... = R_N$.

1.2 Notation and definitions

In this paper, *N-way tensors* (multi-way arrays) are denoted by underlined letters, for example $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is an N-way tensor whose entries are denoted by either of the following ways $\underline{\mathbf{Y}}(i_1, i_2, ..., i_N) = y_{i_1 i_2 \ldots i_N}$. Matrices (*2-way tensors)* are denoted by bold uppercase letters, i.e. $\mathbf{Y} \in \mathbb{R}^{I \times J}$ with entries $\mathbf{Y}(i, j) = y_{ij}$.

An *n-mode fiber* is obtained by fixing all indices except the index in one dimension. For a 3-way (3D) tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I \times J \times K}$ its n-mode fibers are called *column fibers* ($n = 1$), *row fibers* ($n = 2$) and *tube fibers* ($n = 3$)[2].

The operation that transforms a tensor into a matrix is called *matricization* or *unfolding* and consists of arranging the elements of an N-way array into a matrix. Given an N-way tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times J_2 \times \ldots \times I_N}$ we define the *n-mode unfolding matrix* by $\mathbf{Y}_{(n)} \in \mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$, which is obtained by grouping all indices of dimensions $m \neq n$. For example, given a 3-way tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I \times J \times K}$ the corresponding 1-mode unfolding matrix is obtained by creating a new index corresponding to the grouping of indices j and k which is denoted by $\mathbf{Y}_{(1)}(i, (jk)) = y_{ijk}$ (see Fig. 1). It is important to note that unfolding is not unique and different unfolding operations can be defined for N-way tensors, see for example [18, 19, 7] for a formal and general treatment.

We use calligraphic style letters to denote a subset of indices in any dimension, for example, for 3D tensors with indices $i = 1, 2, ..., I$, $j = 1, 2, ..., J$ and $k = 1, 2, ..., K$ we define the subsets containing $P \leq I, J, K$ indices as: $\mathcal{I} = [i_1, i_2, ..., i_P]$, $\mathcal{J} = [j_1, j_2, ..., j_P]$ and $\mathcal{K} = [k_1, k_2, ..., k_P]$. Then we define the *subtensor* $\underline{\mathbf{W}} \in \mathbb{R}^{P \times P \times P}$ as the one determined by indices $\mathcal{I}$, $\mathcal{J}$ and $\mathcal{K}$, i.e. $\underline{\mathbf{W}} = \underline{\mathbf{Y}}(\mathcal{I}, \mathcal{J}, \mathcal{K})$. We also extract matrices containing the tensor fibers determined by these subsets of indices as follows: $\mathbf{C}_1 = \mathbf{Y}_{(1)}(:, \mathcal{J} \times \mathcal{K}) \in \mathbb{R}^{I \times P^2}$, $\mathbf{C}_2 = \mathbf{Y}_{(2)}(:, \mathcal{I} \times \mathcal{K}) \in \mathbb{R}^{J \times P^2}$ and $\mathbf{C}_3 = \mathbf{Y}_{(3)}(:, \mathcal{I} \times \mathcal{J}) \in \mathbb{R}^{K \times P^2}$ for column, row and tube fibers respectively, where ":" denotes all the indices in a dimension and $\mathcal{J} \times \mathcal{K}$ means all indices produced as a combination of an index in $\mathcal{J}$ and an index in $\mathcal{K}$.

The rank-$(R_1, R_2, ..., R_N)$ Tucker model [20, 21] is a tensor decomposition based on a sum of *rank-1* tensors and it is defined as follows:

$$y_{i_1 i_2 \ldots i_N} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \ldots \sum_{r_N=1}^{R_N} g_{r_1 r_2 \ldots r_N} \mathbf{A}_1(i_1, r_1) \mathbf{A}_2(i_2, r_2) \ldots \mathbf{A}_N(i_N, r_N), \quad (2)$$

where $g_{r_1 r_2 \ldots r_N}$ are the entries of the *core tensor* $\underline{\mathbf{G}} \in \mathbb{R}^{R_1 \times R_2 \times \ldots \times R_N}$ and $\mathbf{A}_n \in \mathbb{R}^{I_n \times R_n}$ are the *factor matrices*. Equation (2) can be written in a standard compact form by using the n-mode product "$\times_n$" as follows [22]:

$$\underline{\mathbf{Y}} = \underline{\mathbf{G}} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2 \ldots \times_N \mathbf{A}_N, \quad (3)$$

or following the notation introduced by Kolda in [7]:

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_N]\!], \quad (4)$$

[2]For convenience, for 3-way tensors we use simplified indices $I = I_1$, $J = I_2$ and $K = I_3$.

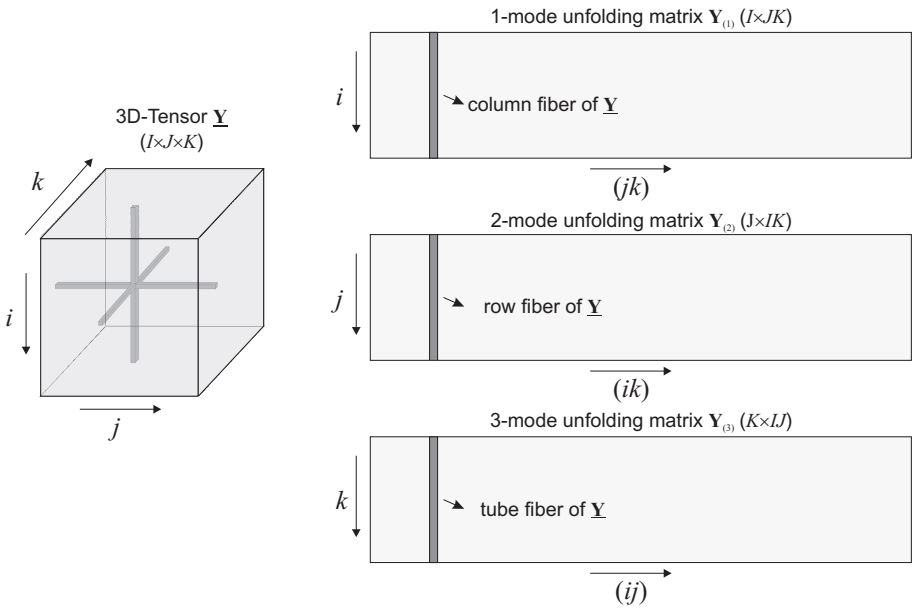

Figure 1: Illustration of a 3-way tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I \times J \times K}$ and its corresponding n-mode unfolding matrices ($n = 1, 2, 3$)

We use the notation of equation (4) to denote the multilinear operation of equation (2) throughout the paper. We note that, in this work, we do not impose orthogonality constraints to the factor matrices as in the HOSVD (high order SVD) model [22, 6]. We also note that, when the core tensor $\underline{\mathbf{G}}$ is diagonal then the Tucker model simplifies to the CP model [23, 24]. For an updated review on tensor decomposition methods and models the reader may refer to [7, 25].

2 Main results

2.1 Matrix case ($N = 2$)

It is known that a pseudo-skeleton representation of a rank-R matrix, in the case of choosing P rows/columns with $P \geq R$, is obtained by using the Moore-Penrose pseudo inverse of the intersection square submatrix $\mathbf{W}$ (instead of the inverse in equation (1)). The following theorem formalizes this case and can be considered as a particular case (exact-representation case) of Theorem 3.2 in [1].

Theorem 1 *Let a matrix $\mathbf{Y} \in \mathbb{R}^{I \times J}$ have rank R and given a selection of P row indices $\mathcal{I} = [i_1, i_2, ..., i_P]$ and P column indices $\mathcal{J} = [j_1, j_2, ..., j_P]$ ($P \geq R$) such that the intersection submatrix $\mathbf{W}$ has also rank R. In this case, the following*

exact CUR representation is obtained:

$$\mathbf{Y} = \mathbf{CUR}, \ \textit{with} \ \mathbf{U} = \mathbf{W}^{\dagger}, \tag{5}$$

where $\mathbf{C} = \mathbf{Y}(:, \mathcal{J})$, $\mathbf{R} = \mathbf{Y}(\mathcal{I}, :)$ *and* $\mathbf{W} = \mathbf{Y}(\mathcal{I}, \mathcal{J})$.

Proof Since matrix $\mathbf{Y}$ has rank R there exist a matrix factorization $\mathbf{Y} = \mathbf{AX}$ with $\mathbf{A} \in \mathbb{R}^{I \times R}$ and $\mathbf{X} \in \mathbb{R}^{R \times J}$. By considering the selected indices $\mathcal{I}$ and $\mathcal{J}$ in this factorization we have:

$$\mathbf{C} = \mathbf{AX}_c, \tag{6}$$
$$\mathbf{R} = \mathbf{A}_r\mathbf{X}, \tag{7}$$
$$\mathbf{W} = \mathbf{A}_r\mathbf{X}_c. \tag{8}$$

with $\mathbf{X}_c = \mathbf{X}(:, \mathcal{J}) \in \mathbb{R}^{R \times P}$ and $\mathbf{A}_r = \mathbf{A}(\mathcal{I}, :) \in \mathbb{R}^{P \times R}$. Since we assume that the intersection matrix $\mathbf{W}$ has rank R and we know that $\text{rank}(\mathbf{A}_r\mathbf{X}_c) \leq \min(\text{rank}(\mathbf{A}_r), \text{rank}(\mathbf{X}_c))$, it follows that $\text{rank}(\mathbf{A}_r) = \text{rank}(\mathbf{X}_c) = R$. Then by multiplying equations (6) and (7) by the Moore-Penrose pseudo-inverses $\mathbf{X}_c^{\dagger}$ and $\mathbf{A}_r^{\dagger}$ we obtain:

$$\mathbf{A} = \mathbf{CX}_c^{\dagger}, \tag{9}$$
$$\mathbf{X} = \mathbf{A}_r^{\dagger}\mathbf{R}. \tag{10}$$

Now, by putting these equations into the factorized form of matrix $\mathbf{Y}$ we finally obtain:

$$\mathbf{Y} = \mathbf{AX} = \mathbf{CX}_c^{\dagger}\mathbf{A}_r^{\dagger}\mathbf{R} = \mathbf{C}(\mathbf{A}_r\mathbf{X}_c)^{\dagger}\mathbf{R} = \mathbf{CW}^{\dagger}\mathbf{R}, \tag{11}$$

where we used the relation $(\mathbf{A}_r\mathbf{X}_c)^{\dagger} = \mathbf{X}_c^{\dagger}\mathbf{A}_r^{\dagger}$ which follows from the fact that matrices $\mathbf{A}_r$, $\mathbf{X}_c$ are full rank.

Note that the result of Theorem 1 is reduced to equation (1) when the number of selected rows/columns is equal to the rank of the matrix $\mathbf{Y}$, i.e. $P = R$. In this case the matrix $\mathbf{W}$ is non-singular.

2.2 Generalization to tensors

In this section we provide two different generalizations of the CUR matrix decomposition which allow us to obtain Tucker models by using only the entries of some selected fibers. We note that, the results of this section are presented for the case of choosing the same number of indices P (or R) in each dimension $n = 1, 2, ..., N$ in order to simplify the notation but it can be easily generalized to the case of using different number of indices in each dimension leading to a non-cubic core tensor.

In the following theorem, our Fiber Sampling Tensor Decomposition (type 1) FSTD1 formula is obtained by extending the steps involved in the proof of Theorem 1 to the case of multi-way arrays. As a result, we obtain an explicit formula that allows us to reconstruct a rank-$(R, R, ..., R)$ Tucker tensor by using a set of P^{N-1} ($P \geq R$) n-mode fibers. First, we present it for the case $N = 3$ in Theorem 2 and next we show its extension to N-way arrays in Theorem 3.

Theorem 2 (FSTD type 1 for 3-way tensors) *Given a tensor* $\underline{\mathbf{Y}} \in \mathbb{R}^{I\times J\times K}$ *having an exact representation by a rank-*(R,R,R) *Tucker model, i.e. there exist matrices* $\mathbf{A}_1 \in \mathbb{R}^{I\times R}$, $\mathbf{A}_2 \in \mathbb{R}^{J\times R}$, $\mathbf{A}_3 \in \mathbb{R}^{K\times R}$ *and a core tensor* $\underline{\mathbf{G}} \in \mathbb{R}^{R\times R\times R}$ *such that*

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!], \tag{12}$$

and, given a selection of P *indices for each dimension* $(P \geq R)$ $\mathcal{I} = [i_1, i_2, ..., i_P]$, $\mathcal{J} = [j_1, j_2, ..., j_P]$ *and* $\mathcal{K} = [k_1, k_2, ..., k_P]$ *such that the unfolding matrices, in all modes, of the intersection subtensor* $\underline{\mathbf{W}} = \underline{\mathbf{Y}}(\mathcal{I}, \mathcal{J}, \mathcal{K})$ *have rank* R*, i.e.*

$$\text{rank}\,(\mathbf{W}_{(1)}) = \text{rank}\,(\mathbf{W}_{(2)}) = \text{rank}\,(\mathbf{W}_{(3)}) = R. \tag{13}$$

In this case, the following exact Tucker representation is obtained:

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{U}}; \mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]\!], \text{ with } \underline{\mathbf{U}} = [\![\underline{\mathbf{W}}; \mathbf{W}_{(1)}^{\dagger}, \mathbf{W}_{(2)}^{\dagger}, \mathbf{W}_{(3)}^{\dagger}]\!], \tag{14}$$

where matrices $\mathbf{C}_1 \in \mathbb{R}^{I\times P^2}$, $\mathbf{C}_2 \in \mathbb{R}^{J\times P^2}$ *and* $\mathbf{C}_3 \in \mathbb{R}^{K\times P^2}$ *are defined by* $\mathbf{C}_1 = \mathbf{Y}_{(1)}(:, \mathcal{J}\times\mathcal{K})$ $\mathbf{C}_2 = \mathbf{Y}_{(2)}(:, \mathcal{I}\times\mathcal{K})$ *and* $\mathbf{C}_3 = \mathbf{Y}_{(3)}(:, \mathcal{I}\times\mathcal{J})$.

Proof Since $\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, its unfolding matrices can be factorized in the following way:

$$\mathbf{Y}_{(1)} = \mathbf{A}_1\mathbf{X}, \tag{15}$$
$$\mathbf{Y}_{(2)} = \mathbf{A}_2\mathbf{V}, \tag{16}$$
$$\mathbf{Y}_{(3)} = \mathbf{A}_3\mathbf{Z}, \tag{17}$$

with $\mathbf{A}_1 \in \mathbb{R}^{I\times P}$, $\mathbf{X} \in \mathbb{R}^{P\times JK}$, $\mathbf{A}_2 \in \mathbb{R}^{J\times P}$, $\mathbf{V} \in \mathbb{R}^{P\times IK}$, $\mathbf{A}_3 \in \mathbb{R}^{K\times P}$, $\mathbf{Z} \in \mathbb{R}^{P\times IJ}$; which follows from looking at the calculation of the entries of the unfolding matrices, for example, for $\mathbf{Y}_{(1)}$ we have that:

$$y_{i(jk)} = \sum_{r_1=1}^{R}\sum_{r_2=1}^{R}\sum_{r_3=1}^{R} \underline{\mathbf{G}}(r_1, r_2, r_3)\mathbf{A}_1(i, r_1)\mathbf{A}_2(j, r_2)\mathbf{A}_3(k, r_3), \tag{18}$$
$$= \sum_{r_1=1}^{R} \mathbf{A}_1(i, r_1) \sum_{r_2=1}^{R}\sum_{r_3=1}^{R} \underline{\mathbf{G}}(r_1, r_2, r_3)\mathbf{A}_2(j, r_2)\mathbf{A}_3(k, r_3), \tag{19}$$
$$= \sum_{r_1=1}^{R} \mathbf{A}_1(i, r_1)\mathbf{X}(r_1, (jk)), \tag{20}$$

where the entries of matrix $\mathbf{X}$ are defined in terms of matrices $\mathbf{A}_2$, $\mathbf{A}_3$ and the core tensor $\underline{\mathbf{G}}$, i.e. $\mathbf{X}(r_1, (jk)) = \sum_{r_2=1}^{R}\sum_{r_3=1}^{R} \underline{\mathbf{G}}(r_1, r_2, r_3)\mathbf{A}_2(j, r_2)\mathbf{A}_3(k, r_3)$.

Therefore, by considering the selected indices $\mathcal{I}$, $\mathcal{J}$ and $\mathcal{K}$ in equations (15-17) we obtain:

$$\mathbf{C}_1 = \mathbf{A}_1\mathbf{X}_c, \tag{21}$$
$$\mathbf{C}_2 = \mathbf{A}_2\mathbf{V}_c, \tag{22}$$
$$\mathbf{C}_3 = \mathbf{A}_3\mathbf{Z}_c. \tag{23}$$

with $\mathbf{X}_c, \mathbf{V}_c, \mathbf{Z}_c \in \mathbb{R}^{R \times P^2}$ being defined by $\mathbf{X}_c = \mathbf{X}(:, \mathcal{J} \times \mathcal{K})$, $\mathbf{V}_c = \mathbf{V}(:, \mathcal{I} \times \mathcal{K})$ and $\mathbf{Z}_c = \mathbf{Z}(:, \mathcal{I} \times \mathcal{J})$. Note also that, the unfolding matrix modes of $\underline{\mathbf{W}}$ have rank R which in turn implies that matrices $\mathbf{X}_c$, $\mathbf{V}_c$ and $\mathbf{Z}_c$ have also rank R. This can be seen, for example for the 1-mode, by observing that $\mathbf{W}_{(1)} = \mathbf{A}_{1r}\mathbf{X}_c$.

By multiplying equations (21-23) by the corresponding Moore-Penrose pseudo-inverses we obtain:

$$\begin{aligned} \mathbf{A}_1 &= \mathbf{C}_1\mathbf{X}_c^{\dagger}, && (24)\\ \mathbf{A}_2 &= \mathbf{C}_2\mathbf{V}_c^{\dagger}, && (25)\\ \mathbf{A}_3 &= \mathbf{C}_3\mathbf{Z}_c^{\dagger}, && (26) \end{aligned}$$

Now, by putting these equations into the Tucker form of the tensor $\underline{\mathbf{Y}}$ we arrive at:

$$\begin{aligned} \underline{\mathbf{Y}} &= [\![\underline{\mathbf{G}}; \mathbf{C}_1\mathbf{X}_c^{\dagger}, \mathbf{C}_2\mathbf{V}_c^{\dagger}, \mathbf{C}_3\mathbf{Z}_c^{\dagger}]\!], && (27)\\ &= [\![\underline{\mathbf{U}}; \mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]\!], && (28) \end{aligned}$$

where the core tensor $\underline{\mathbf{U}} \in \mathbb{R}^{P^2 \times P^2 \times P^2}$ is defined by:

$$\begin{aligned} \underline{\mathbf{U}} &= [\![\underline{\mathbf{G}}; \mathbf{X}_c^{\dagger}, \mathbf{V}_c^{\dagger}, \mathbf{Z}_c^{\dagger}]\!], && (29)\\ &= [\![\underline{\mathbf{G}}; \mathbf{W}_{(1)}^{\dagger}\mathbf{A}_{1r}, \mathbf{W}_{(2)}^{\dagger}\mathbf{A}_{2r}, \mathbf{W}_{(3)}^{\dagger}\mathbf{A}_{3r}]\!], && (30)\\ &= [\![[\![\underline{\mathbf{G}}; \mathbf{A}_{1r}, \mathbf{A}_{2r}, \mathbf{A}_{3r}]\!]; \mathbf{W}_{(1)}^{\dagger}, \mathbf{W}_{(2)}^{\dagger}, \mathbf{W}_{(3)}^{\dagger}]\!], && (31)\\ &= [\![\underline{\mathbf{W}}; \mathbf{W}_{(1)}^{\dagger}, \mathbf{W}_{(2)}^{\dagger}, \mathbf{W}_{(3)}^{\dagger}]\!]. && (32) \end{aligned}$$

Here we have used the following basic property of the Tucker representation: $[\![\underline{\mathbf{G}}; \mathbf{A}_1\mathbf{B}_1, \mathbf{A}_2\mathbf{B}_2, \mathbf{A}_3\mathbf{B}_3]\!] = [\![[\![\underline{\mathbf{G}}; \mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3]\!]; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$.

In the next theorem we extend the result of Theorem 2 to the case of N-way arrays (the proof is omitted since it is easily obtained by following the steps of the previous theorem).

Theorem 3 (FSTD type 1 for N-way tensors) *Given an N-way tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ having an exact representation by a rank-$(R, R, ..., R)$ Tucker model, i.e. there exist a set of matrices $\mathbf{A}_n \in \mathbb{R}^{I_n \times R}$, $n = 1, 2, ..., N$ and a core tensor $\underline{\mathbf{G}} \in \mathbb{R}^{R \times R \times \ldots \times R}$ such that*

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_N]\!], \qquad (33)$$

and, given a selection of P indices for each dimension ($P \geq R$) $\mathcal{I}_n$, $n = 1, 2, ..., N$ such that all the unfolding matrix modes of the intersection subtensor $\underline{\mathbf{W}} = \underline{\mathbf{Y}}(\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_N)$ have rank R. In this case, the following exact tensor representation is obtained:

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{U}}; \mathbf{C}_1, \mathbf{C}_2, ..., \mathbf{C}_N]\!], \ \textit{with}\ \underline{\mathbf{U}} = [\![\underline{\mathbf{W}}; \mathbf{W}_{(1)}^{\dagger}, \mathbf{W}_{(2)}^{\dagger}, ..., \mathbf{W}_{(N)}^{\dagger}]\!], \qquad (34)$$

where matrices $\mathbf{C}_n \in \mathbb{R}^{I_n \times P^{(N-1)}}$ are defined by $\mathbf{C}_n = \mathbf{Y}_{(n)}(:, \bigotimes_{p \neq n} \mathcal{I}_p)$.

Corollary 1 *After applying a basic property of the Tucker model, equation (34) can be written as a rank-$(P, P, ..., P)$ Tucker representation as follows:*

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{W}}; \mathbf{C}_1\mathbf{W}_{(1)}^{\dagger}, \mathbf{C}_2\mathbf{W}_{(2)}^{\dagger}, ..., \mathbf{C}_N\mathbf{W}_{(N)}^{\dagger}]\!]. \tag{35}$$

Remark 1 *Theorem 3 is simplified to the matrix case of Theorem 1 when $N = 2$ since, in this case, we have $\mathbf{C}_1 = \mathbf{C}$, $\mathbf{C}_2 = \mathbf{R}^T$ and the core matrix is $\mathbf{U} = [\![\mathbf{W}; \mathbf{W}_{(1)}^{\dagger}, \mathbf{W}_{(2)}^{\dagger}]\!] = \mathbf{W}^{\dagger}\mathbf{W}\mathbf{W}^{\dagger} = \mathbf{W}^{\dagger}$.*

FSTD1 model in Theorems 2 and 3 is essentially redundant since it uses P^2 ($P \geq R$) n-mode fibers and it is known that an exact Tucker representation exists by using only R n-mode fibers in each mode [2] as factor matrices. In the following theorem, our Fiber Sampling Tensor Decomposition (type 2) FSTD2 formula is obtained which provides such a exact representation by using exactly R n-mode fibers. First, we present it for the case $N = 3$ and we show its extension to N-way arrays at the end of this section (Theorem 5).

Theorem 4 (FSTD type 2 for 3-way tensors) *Given a tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I\times J\times K}$ having an exact representation by a rank-(R, R, R) Tucker model, i.e. there exist matrices $\mathbf{A}_1 \in \mathbb{R}^{I\times R}$, $\mathbf{A}_2 \in \mathbb{R}^{J\times R}$, $\mathbf{A}_3 \in \mathbb{R}^{K\times R}$ and a core tensor $\underline{\mathbf{G}} \in \mathbb{R}^{R\times R\times R}$ such that*

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!], \tag{36}$$

and, given a selection of R row/column/tube fibers arranged as columns of matrices $\mathbf{C}_1 \in \mathbb{R}^{I\times R}$, $\mathbf{C}_2 \in \mathbb{R}^{J\times R}$ and $\mathbf{C}_3 \in \mathbb{R}^{K\times R}$ having all rank-R, and given some sets of R indices $\mathcal{I} = [i_1, i_2, ..., i_R]$, $\mathcal{J} = [j_1, j_2, ..., j_R]$ and $\mathcal{K} = [k_1, k_2, ..., k_R]$ such that the matrices $\mathbf{W}_1 = \mathbf{C}_1(\mathcal{I}, :)$, $\mathbf{W}_2 = \mathbf{C}_2(\mathcal{J}, :)$ and $\mathbf{W}_3 = \mathbf{C}_3(\mathcal{K}, :)$ are non-singular. In this case, the following exact decomposition is obtained:

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{U}}; \mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]\!], \text{ with } \underline{\mathbf{U}} = [\![\underline{\mathbf{W}}; \mathbf{W}_1^{-1}, \mathbf{W}_2^{-1}, \mathbf{W}_3^{-1}]\!], \tag{37}$$

where $\underline{\mathbf{W}} = \underline{\mathbf{Y}}(\mathcal{I}, \mathcal{J}, \mathcal{K}) \in \mathbb{R}^{R\times R\times R}$ is the intersection subtensor.

Proof Since $\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$, its unfolding matrices can be factorized as in equations (15-17). By considering the indices of the selected n-mode fibers in these equations we obtain:

$$\mathbf{C}_1 = \mathbf{A}_1\mathbf{X}_c, \tag{38}$$
$$\mathbf{C}_2 = \mathbf{A}_2\mathbf{V}_c, \tag{39}$$
$$\mathbf{C}_3 = \mathbf{A}_3\mathbf{Z}_c. \tag{40}$$

with $\mathbf{X}_c, \mathbf{V}_c, \mathbf{Z}_c \in \mathbb{R}^{R\times R}$ being non-singular (this fiber selection exists since matrices $\mathbf{C}_1$, $\mathbf{C}_2$ and $\mathbf{C}_2$ are assumed full-rank). Note that this is similar to equations (21-23), but now, only R fibers are considered (instead of P^2).

Now, by multiplying equations (38-40) by the corresponding inverse matrices, we obtain:

$$\mathbf{A}_1 = \mathbf{C}_1\mathbf{X}_c^{-1}, \tag{41}$$
$$\mathbf{A}_2 = \mathbf{C}_2\mathbf{V}_c^{-1}, \tag{42}$$
$$\mathbf{A}_3 = \mathbf{C}_3\mathbf{Z}_c^{-1}. \tag{43}$$

If we consider the above equations evaluated at indices $\mathcal{I}$, $\mathcal{J}$ and $\mathcal{K}$ we obtain:

$$\begin{aligned} \mathbf{A}_1(\mathcal{I},:) &= \mathbf{W}_1\mathbf{X}_c^{-1}, & (44)\\ \mathbf{A}_2(\mathcal{J},:) &= \mathbf{W}_2\mathbf{V}_c^{-1}, & (45)\\ \mathbf{A}_3(\mathcal{K},:) &= \mathbf{W}_3\mathbf{Z}_c^{-1}, & (46) \end{aligned}$$

where $\mathbf{W}_1 = \mathbf{C}_1(\mathcal{I},:)$, $\mathbf{W}_2 = \mathbf{C}_2(\mathcal{J},:)$ and $\mathbf{W}_3 = \mathbf{C}_3(\mathcal{K},:)$ are assumed non-singular. By putting these equations into the Tucker form of the tensor $\underline{\mathbf{Y}}$ we arrive at:

$$\begin{aligned} \underline{\mathbf{Y}} &= [\![\underline{\mathbf{G}}; \mathbf{C}_1\mathbf{X}_c^{-1}, \mathbf{C}_2\mathbf{V}_c^{-1}, \mathbf{C}_3\mathbf{Z}_c^{-1}]\!], & (47)\\ &= [\![\underline{\mathbf{U}}; \mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]\!], & (48) \end{aligned}$$

where the core tensor $\underline{\mathbf{U}} \in \mathbb{R}^{R\times R\times R}$ is defined by:

$$\begin{aligned} \underline{\mathbf{U}} &= [\![\underline{\mathbf{G}}; \mathbf{X}_c^{-1}, \mathbf{V}_c^{-1}, \mathbf{Z}_c^{-1}]\!], & (49)\\ &= [\![\underline{\mathbf{G}}; \mathbf{W}_1^{-1}\mathbf{A}_{1r}, \mathbf{W}_2^{-1}\mathbf{A}_{2r}, \mathbf{W}_3^{-1}\mathbf{A}_{3r}]\!], & (50)\\ &= [\![[\![\underline{\mathbf{G}}; \mathbf{A}_{1r}, \mathbf{A}_{2r}, \mathbf{A}_{3r}]\!]; \mathbf{W}_1^{-1}, \mathbf{W}_2^{-1}, \mathbf{W}_3^{-1}]\!], & (51)\\ &= [\![\underline{\mathbf{W}}; \mathbf{W}_1^{-1}, \mathbf{W}_2^{-1}, \mathbf{W}_3^{-1}]\!], & (52) \end{aligned}$$

where $\underline{\mathbf{W}} = \underline{\mathbf{Y}}(\mathcal{I},\mathcal{J},\mathcal{K}) \in \mathbb{R}^{R\times R\times R}$ is the intersection sub-tensor.

In the next theorem we extend this result to the case of N-way arrays (the proof is omitted since it is easily obtained by following the steps of the previous theorem).

Theorem 5 (FSTD type 2 for N-way tensors) *Given an N-way tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1\times I_2\times\ldots\times I_N}$ having an exact representation by a rank-$(R,R,...,R)$ Tucker model, i.e. there exist a set of matrices $\mathbf{A}_n \in \mathbb{R}^{I_n\times R}$, $n = 1,2,...,N$ and a core tensor $\underline{\mathbf{G}} \in \mathbb{R}^{R\times R\times\ldots\times R}$ such that*

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_N]\!], \qquad (53)$$

and, given a selection of R n-mode fibers arranged as columns of matrices $\mathbf{C}_n \in \mathbb{R}^{I_n\times R}$ $(n = 1,2,...,N)$ having all rank-R, and given some sets of R indices $\mathcal{I}_n$ such that all the matrices $\mathbf{W}_n = \mathbf{C}_n(\mathcal{I}_n,:)$ are non-singular. In this case, the following exact decomposition is obtained:

$$\underline{\mathbf{Y}} = [\![\underline{\mathbf{U}}; \mathbf{C}_1, \mathbf{C}_2, ..., \mathbf{C}_N]\!], \text{ with } \underline{\mathbf{U}} = [\![\underline{\mathbf{W}}; \mathbf{W}_1^{-1}, \mathbf{W}_2^{-1}, ..., \mathbf{W}_N^{-1}]\!], \qquad (54)$$

with $\underline{\mathbf{W}} = \underline{\mathbf{Y}}(\mathcal{I}_1,\mathcal{I}_2,...,\mathcal{I}_N) \in \mathbb{R}^{R\times R\ldots\times R}$.

Remark 2 *It is interesting to note that the FSTD2 model uses not more than $R^N + R\times(I_1+I_2+...+I_N)$ entries of the original Tensor, i.e. not more entries than the ones contained in the core tensor and the factors in its rank-$(R,R,...,R)$ Tensor representation.*

It is noted that the FSTD2 model given in Theorem 5, for $N = 2$, also simplifies to the matrix case of Theorem 1.

3 An Adaptive Algorithm for the FSTD1 model

Theorems 2 and 3 give us a method for an exact calculation of a tensor based on the entries of a subset of fibers (determined by the selected indices in each dimension) when the original data tensor admits an exact rank-$(R, R, ..., R)$ Tucker representation. The Tucker rank R determines a lower bound on the number of indices P to sample in each dimension.

But usually, there is no an exact representation of data in terms of a low order Tucker model and one is often satisfied by finding a good approximation of Tucker rank R [22, 7]. In this case, our FSTD1 formula allows us to obtain an approximation to the tensor data based solely on the information of the selected fibers.

It is important to note that the error in the obtained approximation is sensitive to the selection of indices in each dimension. This phenomenon was analyzed in the matrix case ($N = 2$) for which theoretical bounds of the approximation error are available when the selected rows and columns corresponds to a maximum volume intersection submatrix [1, 5]. The optimal selection of rows and columns subsets in a matrix is a challenging task because one must avoid to test all possible combinations and even the search of a maximum volume submatrix is hard to solve [26]. To alleviate this computational problem there are a class of algorithms known as *cross algorithms* which sequentially selects rows/columns by dynamically finding maximum absolute values within rows/columns of the residuals [27, 28].

However, in the context of the approximation of 3-way tensors, an efficient algorithm to find a 'good enough' submatrix was proposed in [2], later in more detail and with other related topics it was expounded in [16].

We note that, even in the exact-representation case of tensors, we need to develop some technique for the selection of indices to be used in the FSTD1 formula in order to guarantee the assumptions in our theorems. Here we develop an adaptive algorithm that sequentially selects the indices for the case of 3D-way arrays which is inspired on the algorithms described for the matrix case in [27, 28].

Let us consider that we have already selected p, q and r indices in each of the corresponding dimensions, i.e. we define: $\mathcal{I}_p = [i_1, i_2, ..., i_p]$, $\mathcal{J}_q = [j_1, j_2, ..., j_q]$ and $\mathcal{K}_r = [k_1, k_2, ..., k_r]$. We define the approximation-(p, q, r) by using the indices $\mathcal{I}_p$, $\mathcal{J}_q$ and $\mathcal{K}_r$ through the FSTD1 formula (equation (14)) as follows:

$$\widehat{\underline{\mathbf{Y}}}^{(p,q,r)} = [\![\underline{\mathbf{U}}^{(p,q,r)}; \mathbf{C}_1^{(q,r)}, \mathbf{C}_2^{(p,r)}, \mathbf{C}_3^{(p,q)}]\!], \tag{55}$$

where $\underline{\mathbf{U}}^{(p,q,r)}$ is the core tensor (calculated as in equation (14) by using the entries of the intersection subtensor $\underline{\mathbf{W}}^{(p,q,r)} = \underline{\mathbf{Y}}(\mathcal{I}_p, \mathcal{J}_q, \mathcal{K}_r)$), $\mathbf{C}_1^{(q,r)} = \mathbf{Y}_{(1)}(:, \mathcal{J}_q \times \mathcal{K}_r)$, $\mathbf{C}_2^{(p,r)} = \mathbf{Y}_{(2)}(:, \mathcal{I}_p \times \mathcal{K}_r)$ and $\mathbf{C}_3^{(p,q)} = \mathbf{Y}_{(3)}(:, \mathcal{I}_p \times \mathcal{J}_q)$. We define the residual as the difference between the original tensor and its approximation:

$$\underline{\mathbf{E}}^{(p,q,r)} = \underline{\mathbf{Y}} - \widehat{\underline{\mathbf{Y}}}^{(p,q,r)}. \tag{56}$$

Based on the already selected indices $(\mathcal{I}_p, \mathcal{J}_q, \mathcal{K}_r)$ we want to develop a technique for choosing a good new index to be added to the set of selected indices.

First, we note that the residual is exactly zero at the positions of the already selected indices as a consequence of the following proposition. In the following, we will omit giving reference to specific parameters p, q and r in order to simplify the presentation, i.e. $\mathcal{I} \equiv \mathcal{I}_p$, $\widehat{\underline{Y}} \equiv \widehat{\underline{Y}}^{(p,q,r)}$, and so on.

Proposition 1 *Given a selection of $p, q, r \leq R$ indices $\mathcal{I}$, $\mathcal{J}$, $\mathcal{K}$ such that they determine an intersection subtensor with full-rank unfolding matrices $\mathbf{W}_{(n)}$, $n = 1, 2, 3$, then the following equation holds:*

$$\widehat{\underline{\mathbf{Y}}}(\mathcal{I}, \mathcal{J}, \mathcal{K}) = \underline{\mathbf{Y}}(\mathcal{I}, \mathcal{J}, \mathcal{K}). \tag{57}$$

Proof It can be easily proved by evaluating equation (55) at indices ($\mathcal{I}$,$\mathcal{J}$,$\mathcal{K}$) and noting that $\mathbf{C}_1(\mathcal{I}, :) = \mathbf{W}_{(1)}$, $\mathbf{C}_2(\mathcal{J}, :) = \mathbf{W}_{(2)}$ and $\mathbf{C}_3(\mathcal{K}, :) = \mathbf{W}_{(3)}$.

We also note that the residual is zero at the positions determined by indices which, if are selected, they do not increase the rank of the unfolding matrices of the new intersection subtensor. To be more precise, we present the following theoretical result.

Proposition 2 *Let us consider that there exist an index i^*, not included in the current selection ($i^* \notin \mathcal{I}$) such that, restricted to the indices $(\mathcal{J}, \mathcal{K})$, it defines a vector which is a linear combination of a subset of vectors defined by the already selected indices, i.e. there exists a vector $\mathbf{u} \in \mathbb{R}^p$ such that $\mathbf{Y}_{(1)}(i^*, \mathcal{J} \times \mathcal{K}) = \mathbf{u}^T \mathbf{Y}_{(1)}(\mathcal{I}, \mathcal{J} \times \mathcal{K})$. Then the residual at the vector defined by index i^* and the indices $(\mathcal{J}, \mathcal{K})$, is zero:*

$$\widehat{\underline{\mathbf{Y}}}(i^*, \mathcal{J}, \mathcal{K}) = \underline{\mathbf{Y}}(i^*, \mathcal{J}, \mathcal{K}). \tag{58}$$

Proof Let us consider the approximated tensor of equation (55) evaluated at indices $(i^*, \mathcal{J}, \mathcal{K})$ which gives

$$\widehat{\underline{\mathbf{Y}}}(i^*, \mathcal{J}, \mathcal{K}) = [\![\underline{\mathbf{U}}; \mathbf{C}_1(i^*, :), \mathbf{W}_{(2)}, \mathbf{W}_{(3)}]\!]. \tag{59}$$

Now, by noting that $\mathbf{C}_1(i^*, :) = \mathbf{u}^T \mathbf{W}_1$ and expanding the core tensor $\underline{\mathbf{U}} = [\![\underline{\mathbf{W}}; \mathbf{W}_1^\dagger, \mathbf{W}_{(2)}^\dagger, \mathbf{W}_{(3)}^\dagger]\!]$, we finally obtain:

$$\widehat{\underline{\mathbf{Y}}}(i^*, \mathcal{J}, \mathcal{K}) = [\![\underline{\mathbf{W}}; \mathbf{u}^T, \mathbf{I}, \mathbf{I}]\!]. \tag{60}$$

Then, by applying the 1-mode unfolding and by using the linearity hypothesis we arrive at the the desired equation (57)

$$\widehat{\mathbf{Y}}_{(1)}(i^*, \mathcal{J} \times \mathcal{K}) = \mathbf{u}^T \mathbf{W}_{(1)} = \mathbf{Y}_{(1)}(i^*, \mathcal{J} \times \mathcal{K}). \tag{61}$$

At this point, we have enough theoretical results to design our FSTD1 Adaptive Algorithm. The idea is to sequentially add one index in one mode at a time. Let us to analyze, for example, the problem of selecting an additional index i_{p+1} to $\mathcal{I}_p$, i.e. $\mathcal{I}_{(p+1)} = \mathcal{I}_p \cup [i_{p+1}]$. We can pivot on the last selected indices in the other dimensions, i.e. we consider the 1-mode fiber $\underline{\mathbf{Y}}(:, j_q, k_r)$ and we look at the residual in that fiber. We must avoid to select an index with a zero entry in the residual fiber because it may correspond to one of the already selected indices (Proposition 1) or it may produce a non full-rank unfolding matrix in the augmented intersection subtensor (Proposition 2). Therefore, we propose to choose the index corresponding to the maximum absolute value entry in the residual fiber, because, in addition to guarantee the condition of our Theorem 2 (Proposition 2), we make sure that, in the new approximation of the tensor, this non-zero residual entry is canceled (Proposition 1). Additionally, it is reasonable to think that, a large absolute value in the residual at an eligible index means that, if selected, it is likely to contribute to a good reduction in the total error.

A detailed description of our method is shown as Algorithm 1. The idea is to start from an arbitrary random selection of (j_1, k_1) and to find i_1, then by pivoting in (i_1, k_1) we find j_2, with (i_1, j_2) we find k_2 and so on. This procedure is very fast since it only involves finding a maximum value in a single fiber of the residual, i.e. it is not necessary to calculate the residual for the entire tensor. The pivoting technique is illustrated in Figure 2.

Algorithm 1 : `FSTD1 Adaptive Algorithm` **(j_1, k_1, P)**

INPUT: Initial column fiber selection (j_1, k_1) and the number of fibers to be selected P.
OUTPUT: Indices of selected P rows/columns/tubes $\mathcal{I}_P$, $\mathcal{J}_P$ and $\mathcal{K}_P$.

$\mathcal{I}_0 = [\emptyset]$, $\mathcal{J}_1 = [j_1]$, $\mathcal{K}_1 = [k_1]$;
Choose i_1 as the index of Max absolute value in the column fiber $\underline{\mathbf{Y}}(:, j_1, k_1)$;
$\mathcal{I}_1 = \mathcal{I}_0 \cup [i_1]$
$p = 2$;
while $p < P$ **do**
  if $p = 2$ **then**
    Choose j_p as the index of Max absolute value in $\underline{\mathbf{Y}}(i_{p-1}, :, k_{p-1})$; See footnote[3]
  else
    Choose j_p as the index of Max absolute value in $\underline{\mathbf{E}}^{(p-1,p-1,p-1)}(i_{p-1}, :, k_{p-1})$;
  end if
  $\mathcal{J}_p = \mathcal{J}_{p-1} \cup [j_p]$;
  Choose k_p as the index of Max absolute value in $\underline{\mathbf{E}}^{(p-1,p,p-1)}(i_{p-1}, j_p, :)$;
  $\mathcal{K}_p = \mathcal{K}_{p-1} \cup [k_p]$;
  Choose i_p as the index of Max absolute value in $\underline{\mathbf{E}}^{(p-1,p,p)}(:, j_p, k_p)$;
  $\mathcal{I}_p = \mathcal{I}_{p-1} \cup [i_p]$;
  p=p+1;
end while
return $\mathcal{I}_{p-1}$, $\mathcal{J}_{p-1}$, $\mathcal{K}_{p-1}$;

We note that our Algorithm 1 requires to fix the number of desired indices to be used or, which is equivalent, the Tucker rank of the approximation. In

[3] Is easy to show that, when only one index was selected in each mode, i.e. (i_1, j_1, k_1), then the row fiber in the residual defined by $\underline{\mathbf{E}}^{(1,1,1)}(i_1, :, k_1)$ is all zero.

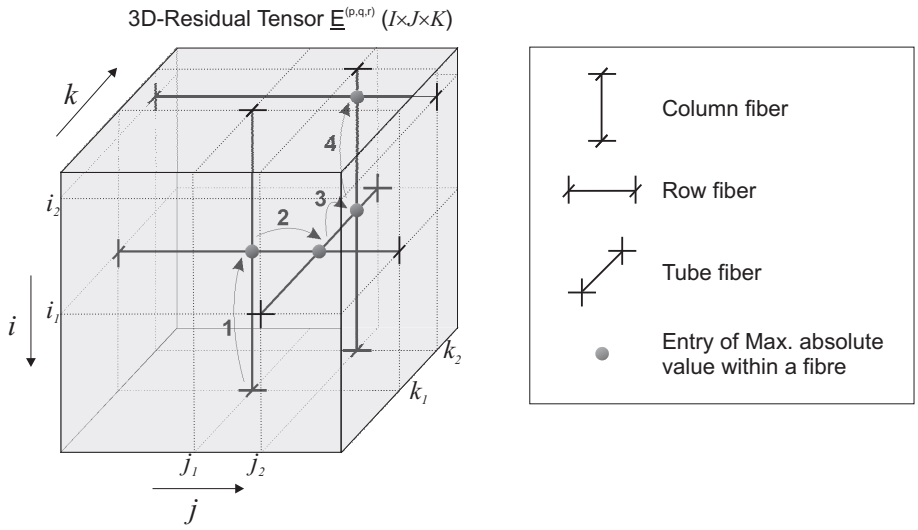

Figure 2: Illustration of the FSTD1 Adaptive Algorithm with sampling of fibers in the 3D residual tensor. Note that we do not need to access to all the entries of the data tensor $\underline{\mathbf{Y}}$.

some applications, this parameter is not known in advance and we may want to stop adding indices when a desirable precision is reached. Of course, we must avoid calculating the total error with equation (56) since it requires to use all the entries of the tensor. Anyway, we can use a practical estimation of the error (stopping criterion) by measuring the norm of the successive correction tensors as proposed in [2].

We also note that our Algorithm 1 (FSTD1 Adaptive Algorithm) is different to the CROSS3D Algorithm proposed in [2]. We do not need to search for the maximum values within slices, our algorithm successively searches for maximum values within fibers and applies equation (14) for computing the core tensor. Additionally, it is highlighted that our algorithm is different to the Tensor-CUR algorithm [13, 14] where the approximation is achieved by applying a CUR algorithm [9] to one unfolding matrix mode by choosing few frontal slices or "slabs" and few tube fibers with a probability based on their ℓ_2 norms which requires to access all the tensor entries.

3.1 Complexity of the FSTD1 Adaptive Algorithm

In this section we analyze the computational complexity of this new algorithm in terms of the size of the tensor I (we assume $I = J = K$) and the Tucker rank of the approximation P.

At every step of the FSTD1 Adaptive Algorithm, a fiber of the current residual tensor is calculated. If we consider that we already selected p indices

in each mode, then the row-fiber determined by the indices (i_p, k_p) is calculated as follows:

$$\underline{\mathbf{E}}^{(p,p,p)}(i_p,:,k_p) = \underline{\mathbf{Y}}(i_p,:,k_p) - [\![\underline{\mathbf{W}}; \mathbf{C}_1(i_p,:)\mathbf{W}^{\dagger}_{(1)}, \mathbf{C}_2\mathbf{W}^{\dagger}_{(2)}, \mathbf{C}_3(k_p,:)\mathbf{W}^{\dagger}_{(3)}]\!], \tag{62}$$

where $\underline{\mathbf{Y}}(i_p,:,k_p) \in \mathbb{R}^{1\times I\times 1}$, $\underline{\mathbf{W}} \in \mathbb{R}^{p\times p\times p}$, $\mathbf{W}_{(n)} \in \mathbb{R}^{p\times p^2}$ $(n = 1, 2, 3)$; $\mathbf{C}_1(i_p,:)$, $\mathbf{C}_3(k_p,:) \in \mathbb{R}^{1\times p^2}$ and $\mathbf{C}_2 \in \mathbb{R}^{I\times p^2}$. Here, $\mathcal{O}(p^4)$ operations are required to compute each $\mathbf{W}^{\dagger}_{(n)}$ $(n = 1, 2, 3)$, $\mathcal{O}(p^3)$ operations for computing the matrix products $\mathbf{C}_1(i_p,:)\mathbf{W}^{\dagger}_{(1)}$ and $\mathbf{C}_3(k_p,:)\mathbf{W}^{\dagger}_{(3)}$, and $\mathcal{O}(Ip^3)$ operations for the product $\mathbf{C}_2\mathbf{W}^{\dagger}_{(2)}$. In addition we have $\mathcal{O}(I(p+p^2+p^3))$ operations for calculating the tensor by matrix product in each mode. Finally, we have additional $\mathcal{O}(I)$ operations to subtract the obtained fiber from the original fiber $\underline{\mathbf{Y}}(i_p,:,k_p)$. Putting all together, we obtain $\mathcal{O}(I(2p^3+p^2+p+1)+3p^4+2p^3) = \mathcal{O}(Ip^3+p^4)$ operations at every step of the algorithm. Therefore, summarizing over $p = 2, 3, ..., P$ we arrive at:

$$\text{COMPLEXITY(Algorithm 1)} = \sum_{p=2}^{P} \mathcal{O}(Ip^3 + p^4) = \mathcal{O}(IP^4 + P^5).$$

First part dominates if I is large and $I >> P$, in this case the total complexity is $\mathcal{O}(P^4 I)$ which is consistent with our experimental results (see Fig. 5). We note that the CROSS3D Algorithm proposed in [2] has theoretical complexity $\mathcal{O}(P^3 I)$.

4 Numerical results

In this section we evaluate the performance of our FSTD1 Adaptive Algorithm by applying it to several types of datasets and by comparing to the results obtained by the Fast-Tucker algorithm available at http://spring.inm.ras.ru/osel, an experimental 3-way implementation of basic ideas for N-way tensors related to those of CROSS3D [2][4]. We also compared our results to the ones obtained by the classical Alternated Least Squares (ALS) method for small tensors ($I = J = K = 100$). Additionally, we provide extensive simulations for very large tensors (up to $I = J = K = 2^{16}$) which allowed us to verify the robustness and asymptotic complexity of our FSTD1 Adaptive Algorithm. All computations were performed with MATLAB Rel. 2007 on a desktop computer equipped with an Intel Core Quad CPU (2.66GHz) with 4GB memory.

4.1 Exact Representation Case

Here we present the results of applying our FSTD1 Adaptive Algorithm to rank-(R, R, R) Tucker tensors $\underline{\mathbf{Y}} \in \mathbb{R}^{I\times J\times K}$ with $I = J = K = 100$ (eq. (4)) by

[4] According to the authors remark, Fast-Tucker Algorithm is formally different from CROSS3D Algorithm and does not take care of optimal performance when $N = 3$.

Table 1: Relative Approximation Error of FSTD1 Adaptive Algorithm for the exact-representation case of rank-(R, R, R) Tucker tensors generated at random.

Rank $P = R$	10	20	30	40	50	60	70	80
Relative Error	5.5E-15	5.9E-15	6E-15	6.9E-15	8.3E-15	8.5E-15	8.8E-15	8.9E-15

randomly generating matrices $\mathbf{A}_n \in \mathbb{R}^{I\times R}$ $(n = 1, 2, 3)$ and core tensors $\underline{\mathbf{G}} \in \mathbb{R}^{R\times R\times R}$ for $R = 10, 20, ..., 80$. The entries were generated by using Gaussian independent identically distributed variables with zero mean and the resulting tensor $\underline{\mathbf{Y}}$ was normalized i.e. $\underline{\mathbf{Y}} \leftarrow \underline{\mathbf{Y}}/\|\underline{\mathbf{Y}}\|_F$ where the Frobenius norm is defined by $\|\underline{\mathbf{Y}}\|_F = \sum_{ijk} y_{ijk}^2$.

In Table 1, the results of applying our FSTD1 Adaptive Algorithm with $P = R$ is shown in terms of the achieved relative error which is defined as $e = \|\underline{\mathbf{Y}} - \widehat{\underline{\mathbf{Y}}}\|_F/\|\underline{\mathbf{Y}}\|_F$. The results were averaged over a total of 50 Monte Carlo (MC) simulations. The achieved errors were under 10^{-14} which is a very good approximation compared to the maximal attainable machine precision EPS$\approx 10^{-16}$.

4.2 Approximate Representation Examples

In order to analyze the case of applying our FSTD1 model to the approximate case, i.e. when we select less indices than the Tucker rank $(P < R)$, we have analyzed the following datasets:

- Random rank-(R, R, R) Tucker tensors: as in the dataset used in the previous section, we have generated 3D tensor data $\underline{\mathbf{Y}} \in \mathbb{R}^{I\times J\times K}$ with $I = J = K = 100$ by randomly generating matrices $\mathbf{A}_n \in \mathbb{R}^{I\times R}$ $(n = 1, 2, 3)$ and a core tensor $\underline{\mathbf{G}} \in \mathbb{R}^{R\times R\times R}$.

- Data tensors generated by sampling a family of smooth functions as follows:
$$\underline{\mathbf{Y}}(i, j, k) = 1/(i^s + j^s + k^s)^{1/s}, \tag{63}$$
with $s \in \mathbb{N}$. These type of tensors were subject of study in [2] for the case $s = 1, 2$.

Given a selection of P indices in each dimension, we compute the rank-(P, P, P) FSTD1 approximation of the generated data tensor as follows:

$$\widehat{\underline{\mathbf{Y}}} = [\![\underline{\mathbf{W}}; \mathbf{C}_1\mathbf{W}_{(1)}^{\dagger}, \mathbf{C}_2\mathbf{W}_{(2)}^{\dagger}, \mathbf{C}_3\mathbf{W}_{(3)}^{\dagger}]\!], \tag{64}$$

where matrices $\mathbf{C}_1 \in \mathbb{R}^{I\times P^2}$, $\mathbf{C}_2 \in \mathbb{R}^{J\times P^2}$ and $\mathbf{C}_3 \in \mathbb{R}^{K\times P^2}$ correspond to column, row and tube fibers and $\underline{\mathbf{W}} \in \mathbb{R}^{P\times P\times P}$ is the intersection subtensor.

In Figure 3 the resulting errors (averaged over a total of 50 MC runs) of two FSTD1 based approximation methods, applied to a rank-$(25, 25, 25)$ random Tucker tensor (a), and applied to a 3-way tensor generated from a smooth function (b), are shown versus the number of selected indices in each dimension

$P = 3, 4, ..., 25$. We have considered two strategies for the selection of indices: 1) Random selection (choosing indices with equal probability) and 2) by applying the FSTD1 Adaptive Algorithm (Algorithm 1). In this figure, it is also shown, as a reference, the error of applying a Tucker Alternated Least Squares (ALS) algorithm by considering a rank-(P, P, P) ($P = 3, 4, ...25$) (we have used the MATLAB Tensor Toolbox [29]). Of course, all three approximating methods are accurate (nearly zero error) when $P = R$ indices are selected as stated by our theorems. However, by choosing the fibers according to the FSTD1 Adaptive Algorithm , we achieve much better results (lower error) than the case of random sampling of indices, specially for the case of the tensor generated by a smooth function (see Fig. 3.(b)).

In the following examples, we present additional results for cases with highly structured datasets where the random selection of indices should be avoided and a more intelligent selection method, such as the case of our FSTD1 Adaptive Algorithm, the Fast-Tucker or the CROSS3D Algorithm [2] must be used.

In Figure 4, we compare our FSTD1 Adaptive Algorithm against the Fast-Tucker algorithm [5]. We applied these two algorithms to a dataset generated according to equation (63) for $s = 1, 2, 3$ and considering $I = J = K = 100$. We proceeded by applying first the Fast-Tucker algorithm setting the accuracy parameter within the range $(10^{-2} - 10^{-9})$ which determines a resulting rank-(P, P, P) Tucker tensor. Then we applied our FSTD1 Adaptive Algorithm by setting the parameter P according to the previous result. We have compared both algorithms in terms of the achieved relative error (Figures 4 (a), (c) and (e)) and in terms of the required computation times (Figures 4 (b), (d) and (f)). FSTD1 Adaptive Algorithm is always faster than Fast-Tucker Algorithm. For $s = 1$ FSTD1 Adaptive Algorithm provides a slightly higher error and for $s = 2, 3$ there is critical number of indices ($P = 11$) and above that value FSTD1 Adaptive Algorithm outperforms the Fast-Tucker algorithm.

4.3 Approximation of large tensors: experimental verification of the FSTD1 Adaptive Algorithm complexity

In this section, we provide additional experimental results of applying our FSTD1 Adaptive Algorithm to tensors generated by the smooth function of eq. (63) with $I = J = K$ and $s = 1$. We put special emphasis in the analysis of the results for very large 3-way tensors (up to $I = 2^{16}$) and for large number of indices (up to $P = 256$).

We have performed simulations for a range of parameters I (size) and P (rank) similar to the one used in the experiments reported with the CROSS3D Algorithm in [2], i.e. $P = 6, 8, ..., 26$ and $I = 2^6, 2^7, ..., 2^{16}$. We note that, it was not possible to perform a formal comparison to the CROSS3D Algorithm since its code is not available in public domain, and the results reported in [2] were obtained with a different hardware and software platform.

In Table 2, the accuracy of the approximation obtained by applying our

[5] http://spring.inm.ras.ru/osel

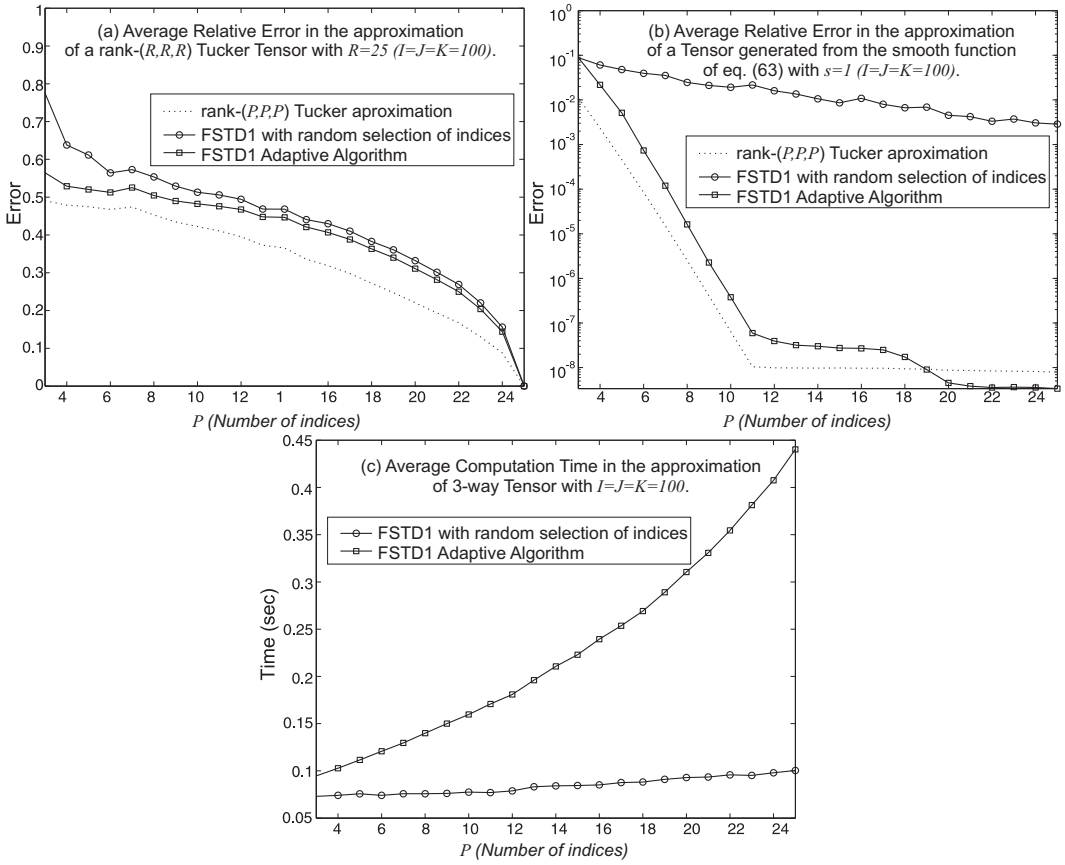

Figure 3: Numerical simulation results of applying our FSTD1 model to the following 3-way tensors ($I = J = K = 100$): a) A randomly generated rank-$(25, 25, 25)$ Tucker tensor, and b) a tensor generated from the smooth function given by eq. (63) with $s = 1$. $P = 3, 4, ..., 25$ indices were selected at random (FSTD1 with random selection) or by our Algorithm 1 (FSTD1 Adaptive Algorithm). A comparison with the error achieved by the ALS Tucker algorithm (MATLAB Tensor Toolbox [29]) is also shown in each case. The results were averaged over a total of 50 MC runs.

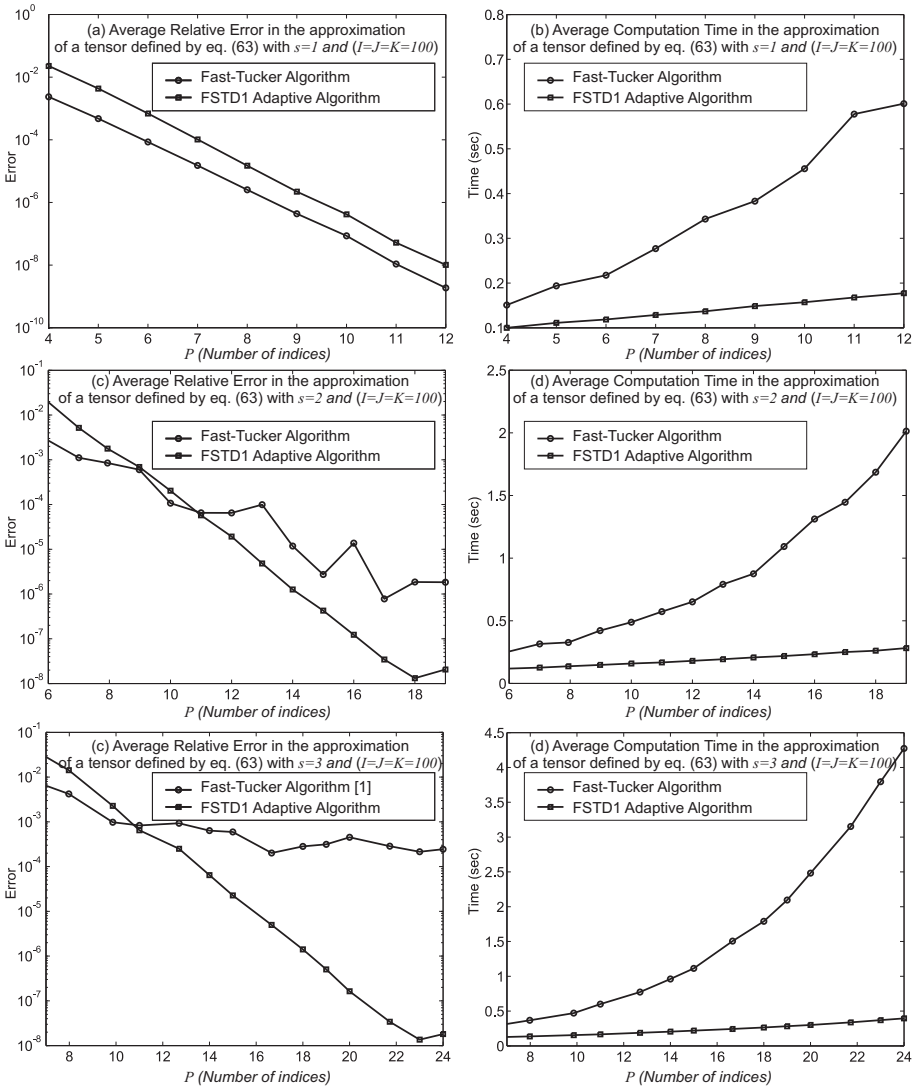

Figure 4: Average Relative Error (left) and Computation Time (right) in the approximation of a 3D tensor $\underline{\mathbf{Y}}(i,j,k) = 1/(i^s + j^s + k^s)^{1/s}$ for $s = 1$ (top), $s = 2$ (middle) and $s = 3$ (bottom). A comparison between our FSTD1 Adaptive Algorithm and the Fast-Tucker Algorithm is shown. The results were averaged over a total of 50 MC runs.

Table 2: Accuracy of the approximation of a large-scale tensor generated by the smooth function given by eq. (63) with $s = 1$. Missing values correspond to cases that require the sampling of too many entries for estimating the relative error.

		P					
		6	8	10	12	14	16
I	2^6	9.06E-05	7.73E-07	1.23E-08	4.67E-09	2.68E-09	1.87E-09
	2^7	1.76E-04	4.59E-06	1.11E-07	7.75E-09	2.34E-09	1.78E-09
	2^8	4.52E-04	1.05E-05	6.27E-07	2.52E-08	3.36E-09	2.10E-09
	2^9	5.22E-04	4.14E-05	1.80E-06	1.11E-07	9.22E-09	2.70E-09
	2^{10}	-	8.21E-05	3.40E-06	3.16E-07	2.27E-08	4.20E-09
	2^{11}	-	1.41E-04	7.61E-06	1.26E-06	6.29E-08	1.19E-08
	2^{12}	-	1.51E-04	1.50E-05	1.20E-06	1.86E-07	1.81E-08
	2^{13}	-	1.15E-04	2.85E-05	2.82E-06	3.79E-07	4.61E-08
	2^{14}	-	-	3.79E-05	4.03E-06	6.54E-07	1.09E-07
	2^{15}	-	-	3.59E-05	5.67E-06	1.44E-06	1.85E-07
	2^{16}	-	-	4.35E-05	8.49E-06	1.82E-06	3.06E-07

		P				
		18	20	22	24	26
I	2^6	1.68E-09	1.77E-09	1.73E-09	1.63E-09	1.86E-09
	2^7	1.65E-09	1.68E-09	1.74E-09	1.95E-09	2.13E-09
	2^8	1.96E-09	1.91E-09	2.09E-09	2.35E-09	2.55E-09
	2^9	2.19E-09	2.19E-09	2.27E-09	2.59E-09	2.75E-09
	2^{10}	2.62E-09	2.58E-09	2.75E-09	2.46E-09	3.06E-09
	2^{11}	3.46E-09	2.77E-09	2.73E-09	2.41E-09	2.81E-09
	2^{12}	3.99E-09	3.53E-09	2.52E-09	2.40E-09	2.36E-09
	2^{13}	8.19E-09	4.28E-09	2.97E-09	2.61E-09	2.51E-09
	2^{14}	1.35E-08	6.20E-09	4.30E-09	3.25E-09	2.37E-09
	2^{15}	2.64E-08	7.33E-09	5.26E-09	4.88E-09	3.72E-09
	2^{16}	5.20E-08	1.17E-08	6.26E-09	5.64E-09	5.59E-09

FSTD1 Adaptive Algorithm is shown for a wide range of parameters I and P. As in [2], the accuracy of the approximation was computed by sampling the elements of the array, since it is not possible to check all the elements for large I. The size of the sample was determined by the following rule: if the sample was doubled, the estimated error should change no more than 10%. As it can be seen, the approximation method is robust providing similar accuracies to the ones reported in [2].

In Figs. 5-(a) and 5-(b) we present the obtained computation times versus the number of indices P (for fixed I) and versus size I (for fixed P) respectively. From this figure we confirm the asymptotic theoretical complexity $\mathcal{O}(P^4 I)$ which is observed for $P > 32$ in Fig. 5-(a) and for $I > 2000$ approximately. It is important to highlight that this algorithm is very fast, for example, the approximation of a huge tensor with $I = 2^{16}$ (which would require 2 Peta Bytes of memory) is

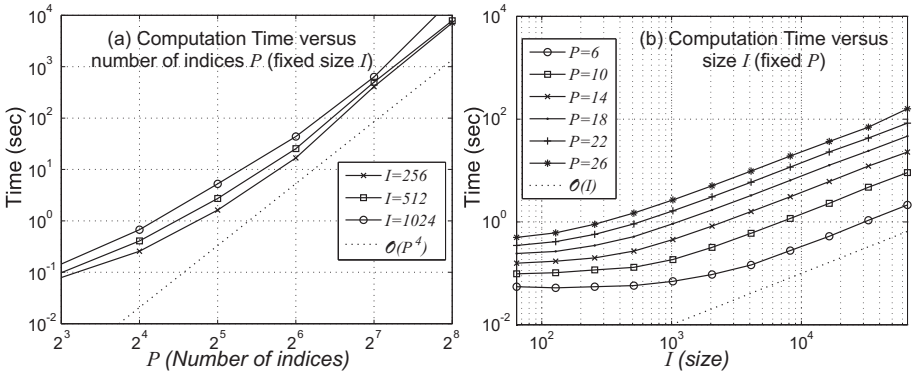

Figure 5: FSTD1 Adaptive Algorithm: Computation Time versus number of indices P and size I in the approximation of a 3D tensor $\underline{\mathbf{Y}}(i,j,k) = 1/(i+j+k)$. $\mathcal{O}(P^4)$ and $\mathcal{O}(I)$ lines are shown as a reference for the asymptotic complexity.

computed in only 158 seconds by using $P = 26$ indices in each mode.

5 Conclusions

We have provided two generalizations of the CUR matrix decomposition to the case of N-way tensors allowing us to reconstruct a whole data tensor based solely on a subset of its entries. Based on our FSTD1 model, we developed a simple adaptive algorithm for a sequential selection of indices. This decomposition is very attractive for dealing with large-size data sets because: I) It is not required to have access to all the data entries for computing its approximation, and II) the selection of the indices to sample in each dimension can be done quickly by successively finding maximum values within fibers of residuals. Our experimental results showed that our algorithm is faster than the Fast-Tucker algorithm and can achieve better results in terms of the obtained error for some particular datasets. The computational complexity of our FSTD1 Adaptive Algorithm is $\mathcal{O}(P^4 I)$ and the complexity for CROSS3D Algorithm introduced in [2] is $\mathcal{O}(P^3 I)$. Hence, our algorithm is attractive for low-rank approximation of tensors since it provides very fast computations for moderate ($P << I$) as our simulation results shows. For example, for $I = 2^{16}$ and $P = 26$ we obtained a very precise approximation in only 158 seconds[6].

[6]The codes of CROSS3D[2] are not in the public domain, so the same-style comparison with CROSS3D was not possible.

Acknowledgments

We thank the Editor Professor Eugene Tyrtyshnikov and anonymous Reviewers for their very useful comments, questions and suggestions that allowed us to improve considerably the paper.

References

[1] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra and its Applications*, 261:1–21, 1997.

[2] I. V. Oseledets, D. V. Savostianov, and E. E. Tyrtyshnikov. Tucker dimensionality reduction of three-dimensional arrays in linear time. *SIAM Journal on Matrix Analysis and Applications*, 30(3):939–956, 2008.

[3] F. R. Gantmacher. *Theory of Matrices*. Chelsea, New York, 1959.

[4] S. A. Goreinov and E. E. Tyrtyshnikov. The maximal-volume concept in approximation by low-rank matrices. *Contemporary Mathematics*, 280:47–51, 2001.

[5] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. Pseudoskeleton approximations by matrices of maximal volume. *Mathematical Notes*, 62(4):515–519, 1997.

[6] L. De Lathauwer, B. De Moor, and J. Vandewalle. On the best rank-1 and rank- (R1,R2,..,Rn) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21:1324–1342, 2000.

[7] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, September 2009.

[8] G. W. Stewart. Four algorithms for the the efficient computation of truncated pivoted QR approximations to a sparse matrix. *Numerische Mathematik*, 83(2):313–323, 1999.

[9] P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36:184–206, 2006.

[10] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, May 2008.

[11] J. Sun, Y. Xie, H. Zhang, and C. Faloutsos. Less is more: Compact matrix decomposition for large sparse graphs. In *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)*, 2007.

[12] M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, January 2009.

[13] M. W. Mahoney, M. Maggioni, and P. Drineas. Tensor-CUR decompositions for tensor-based data. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 327–336, New York, NY, USA, 2006. ACM.

[14] M. W. Mahoney, M. Maggioni, and P. Drineas. Tensor-CUR decompositions for tensor-based data. *SIAM Journal on Matrix Analysis and Applications*, 30(3):957–987, 2008.

[15] E. E. Tyrtyshnikov. Low-rank structures and tensor approximations for huge-size data sets. In *Proceedings of the 8th Hellenic European Research on Computer Mathematics and its Applications Conferences - HERCMA 2007*, pages 1–9, Athens, 2007.

[16] S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. How to find a good submatrix. Research Report 08-12, ICM, www.math.hkbu.edu.hk/ICM, 2008.

[17] S. A. Goreinov. On cross approximation of multi-index arrays. *Doklady Mathematics*, 77(3):404–406, July 2008.

[18] I. V. Oseledets and E. E. Tyrtyshnikov. Breaking the curse od dimensionality, or how to use SVD in many dimensions. *SIAM J. Sci. Comput.*, 31(5):3744–3759, 2009.

[19] I. V. Oseledets. Compact matrix form of the d-dimensional tensor decomposition. Submitted to: SIAM Journal of Scientific Computing, 2009.

[20] L. R. Tucker. Implications of factor analysis of three-way matrices for measurement of change. In C. W. Harris, editor, *Problems in Measuring Change*, pages 122–137. University of Wisconsin Press, 1963.

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

[22] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21:1253–1278, 2000.

[23] J. D. Carroll and J. J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35:283–319, 1970.

[24] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "exploratory" multi-modal factor analysis. *UCLA working papers in phonetics*, 16:1–84, 1970.

[25] A. Cichocki, R. Zdunek, A. H. Phan, and S. I. Amari. *Nonnegative Matrix and Tensor Factorizations*. Wiley, 2009.

[26] J. J. Bartholdi. A good submatrix is hard to find. *Operations Research Letters*, 1(5):190–193, November 1982.

[27] E. E. Tyrtyshnikov. Incomplete cross approximation in the mosaic-skeleton method. *Computing*, 64(4):367–380, 2000.

[28] J. M. Ford and E. E. Tyrtyshnikov. Combining Kronecker product approximation with discrete wavelet transforms to solve dense, function-related systems. *SIAM J. Sci. Comput.*, 25:961–981, 2003.

[29] B. W. Bader and T. G. Kolda. Matlab tensor toolbox version 2.2. http://csmr.ca.sandia.gov/ tgkolda/TensorToolbox/, January 2007.