ADAPTIVE MULTILINEAR SVD FOR STRUCTURED TENSORS

Rémy Boyer

Laboratoire des Signaux et Systèmes (LSS)
CNRS, Université Paris XI (UPS), SUPELEC
Gif-Sur-Yvette, France
remy.boyer@lss.supelec.fr

Roland Badeau

GET - Télécom Paris, dept. TSI
46 rue Barrault
75634 Paris Cedex 13 France
roland.badeau@enst.fr

ABSTRACT

The Higher-Order SVD (HOSVD) is a generalization of the SVD to higher-order tensors (*ie.* arrays with more than two indexes) and plays an important role in various domains. Unfortunately, the computational cost of this decomposition is very high since the basic HOSVD algorithm involves the computation of the SVD of three *highly redundant block-Hankel* matrices, called modes. In this paper, we present an ultra-fast way of computing the HOSVD of a third-order structured tensor. The key result of this work lies in the fact it is possible to reduce the basic HOSVD algorithm to the computation of the SVD of three *non-redundant Hankel* matrices whose columns are multiplied by a given weighting function. Next, we exploit an FFT-based implementation of the orthogonal iteration algorithm in an adaptive way. Even though for a square $(I \times I \times I)$ tensor the complexity of the basic full-HOSVD is $O(I^4)$ and $O(rI^3)$ for its r-truncated version, our approach reaches a linear complexity of $O(rI\log_2(I))$.

1. INTRODUCTION

The subject of multilinear decomposition is now mature [1,2]. There are essentially two families. The first one is known under the name of CANDECOMP/PARAFAC (CANonical DECOMPosition or PARAllel FACtors model) and was independently proposed in [6, 7]. This decomposition is very useful in several applications and is relied to the tensor rank [9]. The second one is related to the multidimensional rank [8] and is known under the name of Tucker decomposition [5]. This decomposition is a more general form which is often used. Orthogonality constraints are not required in the general Tucker decomposition but if needed one can refer to the Higher-Order Singular Value Decomposition (HOSVD) [8] or multilinear SVD.

The HOSVD is a generalization of the SVD to higher-order tensors (*ie.* arrays with more than two indexes). This decomposition plays an important role in various domains, such as harmonic retrieval [3], image processing, telecommunications, biomedical applications (magnetic resonance imaging and electrocardiography), web search [13], computer facial recognition [11], handwriting analysis [12], statistical analysis by Independent Component Analysis (ICA) [8].

In [8], it has been shown that the HOSVD of a third-order tensor involves the computation of the SVD of three matrices called modes. As a consequence, the computational cost of this algorithm is very high. In the case of structured tensors, these modes are highly redundant matrices (*ie.* many columns are repeated). In this paper, we propose a ultra-fast implementation of the r-truncated HOSVD for structured tensors which takes the redundant structure of each mode into account. The key result of this work lies in the fact that it is possible to reduce the basic HOSVD algorithm to the computation of the SVD of three non-redundant (all columns are different) Hankel matrices whose columns are multiplied by a given weighting function. Next, we exploit the *orthogonal iteration* algorithm in an adaptive way. Even though for square $(I \times I \times I)$ tensors the complexity of the basic full-HOSVD is $O(I^4)$ and $O(rI^3)$ for its r-truncated version, our approach reaches a linear complexity of $O(rI\log_2(I))$.

2. PRELIMINARIES IN MULTILINEAR ALGEBRA

2.1. Mode of a tensor

There are several ways to represent a $I_1 \times I_2 \times I_3$ third-order complex-valued tensor $\mathcal{A}$ as a collection of matrices.

Definition 1 *We define the modes (also called "matrix unfoldings") A_1, A_2, A_3 as follows:*

$$[A_1]_{i_1,(i_3-1)I_3+i_2} = [\mathcal{A}]_{i_1 i_2 i_3}, \quad (1)$$

$$[A_2]_{i_2,(i_3-1)I_3+i_1} = [\mathcal{A}]_{i_1 i_2 i_3}, \quad (2)$$

$$[A_3]_{i_3,(i_1-1)I_1+i_2} = [\mathcal{A}]_{i_1 i_2 i_3}. \quad (3)$$

where we have denoted the entries of $\mathcal{A}$ by $[\mathcal{A}]_{i_1 i_2 i_3}$ with $i_s \in [0 : I_s - 1]$. These matrices are of dimension $(I_1 \times I_2 I_3)$, $(I_2 \times I_3 I_1)$, $(I_3 \times I_1 I_2)$, respectively.

The dimensions of the vector spaces generated by the columns of the modes of $\mathcal{A}$ are called column rank (or 1-mode rank) R_1, row rank (or 2-mode rank) R_2 and 3-mode rank R_3, respectively.

Definition 2 *A third-order tensor whose s-mode (or multidimensional) rank is equal to R_s, $s \in [1 : 3]$, is called a rank-(R_1, R_2, R_3) tensor.*

2.2. Multilinear SVD (HOSVD)

Theorem 1 (Third-Order SVD [5,8]) *Every $I_1 \times I_2 \times I_3$ tensor $\mathcal{A}$ can be written as the product:*

$$\mathcal{A} = \mathcal{S} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)} \quad (4)$$

in which $\times_s$ represents the Tucker s-mode product [8], $U^{(s)}$ is an unitary $I_s \times I_s$ matrix and $\mathcal{S}$ is an all-orthogonal and ordered $I_1 \times I_2 \times I_3$ tensor. All-orthogonality means that the matrices $S_{i_s=\alpha}$, obtained by fixing the s-th index to α, are mutually orthogonal w.r.t. the standard inner product. Ordering means that $\|S_{i_s=0}\| \geqslant \|S_{i_s=1}\| \geqslant \ldots \geqslant \|S_{i_s=I_s-1}\| \geqslant 0$ for all possible values of s. The Frobenius-norms

$\|S_{i_s=i}\|$, symbolized by $\sigma_i^{(s)}$, are the s-mode singular values of $\mathcal{A}$ and the columns of $U^{(s)}$ are the s-mode singular factors.

This decomposition is a generalization of the SVD because the diagonality of the matrix containing the singular values, in the matrix case, is a special case of all-orthogonality. Also, the HOSVD of a second-order tensor (matrix) yields the matrix SVD, up to trivial indeterminacies. The matrix of s-mode singular factors, $U^{(s)}$, can be found as the matrix of left singular factors of the mode A_s, defined in (1)–(3). The s-mode singular values correspond to the singular values of this matrix unfolding. Note that the s-mode singular factors of a tensor, corresponding to the nonzero s-mode singular values, form an orthonormal basis for its s-mode vector subspace, like in the matrix case.

The core tensor $\mathcal{S}$ can then be computed (if needed) by bringing the matrices of s-mode singular factors to the left side of equation (4):

$$\mathcal{S} = \mathcal{A} \times_1 U^{(1)^H} \times_2 U^{(2)^H} \times_3 U^{(3)^H} \tag{5}$$

where $(.)^H$ denotes the complex conjugate.

2.3. Mode decompositions

Expression (4) can be written in terms of modes as follows:

$$\begin{aligned} A_1 &= U^{(1)} S_1 \left(U^{(3)} \otimes U^{(2)}\right)^T, \\ A_2 &= U^{(2)} S_2 \left(U^{(3)} \otimes U^{(1)}\right)^T, \\ A_3 &= U^{(3)} S_3 \left(U^{(1)} \otimes U^{(2)}\right)^T, \end{aligned}$$

in which $\otimes$ denotes the Kronecker product and S_1, S_2 and S_3 denote respectively the first, second and third mode of the core tensor $\mathcal{S}$. Note that the columns of $U^{(s)}$ span the space generated by the s-mode vectors of $\mathcal{A}$.

3. COMPLEXITY OF FULL AND TRUNCATED HOSVD

Let $N = I_1 + I_2 + I_3 - 1$, $Q = I_1 I_2 I_3$ and $I = \left[\frac{N+1}{3}\right]$. The computational costs of the various algorithms presented below are related to the flop (*floating point operation*) count. For example, a dot product of I-dimensional vectors involves $2I$ flops (I multiplications plus I additions).

3.1. Computation of the full basic HOSVD

The calculation of the full basic HOSVD of tensor $\mathcal{A}$ requires the computation, for all $s \in [1:3]$, of the left factor $U^{(s)}$ in the full SVD of matrix A_s, as defined above. Once the three matrices $U^{(s)}$ have been obtained, the tensor $\mathcal{S}$ can be computed by means of the Tucker product

$$\mathcal{S} = \mathcal{A} \times_1 U^{(1)^H} \times_2 U^{(2)^H} \times_3 U^{(3)^H}. \tag{6}$$

According to [4, pp. 253–254], the average computational cost of the full SVD of an $n \times l$ matrix with $n < l$ is $k_{\mathrm{f}} n^2 l$ flops, where the constant k_{f} depends on the SVD algorithm (*e.g.* Golub-Reinsch or R-SVD). Here, only the left term of the SVD needs to be computed, which can be achieved faster than a complete SVD (*i.e.* with a lower value of k_{f}). Besides, the matrix A_1 has $n = I_1$ rows and $l = I_2 I_3$ columns. Assuming that $I_2 I_3$ is often greater than I_1, the computational cost of the SVD of A_1 is $k_{\mathrm{f}} I_1^2 I_2 I_3$ flops.

The computational costs of the full HOSVD are summarized in table 1. In particular, the maximal complexity over all values of I_1, I_2, I_3 satisfying $N = I_1 + I_2 + I_3 - 1$ is obtained for square tensors ($I_1 = I_2 = I_3 = I$) and equals $(3k_f + 6)I^4$.

3.2. Computation of the truncated HOSVD

In many applications, we are interested in computing the HOSVD truncated at orders (r_1, r_2, r_3) (r_s is often supposed to be much lower than I_s). Let $r = \frac{1}{3}(r_1 + r_2 + r_3)$. This truncated HOSVD can be obtained in the same way as the full HOSVD, except that the three SVD involved in its computation are truncated at orders r_s. Besides, the truncated SVD of an $n \times l$ matrix can be computed faster than its full SVD, by means of the orthogonal iteration method [4, pp. 410–411] for instance. The average computational cost of this truncated SVD is $k_{\mathrm{t}} r_s n l$, where the constant k_{t} depends on the algorithm (k_{t} is generally greater than k_{f}). Therefore, applied to matrix A_s, its cost is $O(r_s I^3)$ flops for square tensors. Note that we assume (1) $I_s \gg \log_2(I_s)$ and (2) $\log_2(I_s)$ is the same order as r_s. The computational costs of the r-truncated HOSVD are summarized in table 1.

Table 1. Full and r-truncated HOSVD Algorithms

Operation	Full	r-truncated HOSVD
SVD of A_1	$k_{\mathrm{f}} I_1 Q$	$k_{\mathrm{t}} r_1 Q$
SVD of A_2	$k_{\mathrm{f}} I_2 Q$	$k_{\mathrm{t}} r_2 Q$
SVD of A_3	$k_{\mathrm{f}} I_3 Q$	$k_{\mathrm{t}} r_3 Q$
Tucker product	$6IQ$	$6rQ$
Total	$(3k_{\mathrm{f}} + 6)IQ$	$(3k_{\mathrm{t}} + 6)rQ$

4. FAST ALGORITHMS FOR HANKEL-STRUCTURED TENSORS

This section is dedicated to Hankel-structured tensors but other structures as for instance circulant matrices can be exploited in a similar way.

4.1. Definition of an Hankel-structured tensor

A third-order $I_1 \times I_2 \times I_3$ Hankel-structured tensor is defined according to:

$$[\mathcal{H}]_{i_1 i_2 i_3} = x[i_1 + i_2 + i_3] \tag{7}$$

where $x[n]$ is the n-th sample of the analyzed signal and $i_s \in [0 : I_s - 1]$ for $s \in [1:3]$.

4.2. Compressed and windowed modes

The structure of the considered tensor induces a strong redundancy in its modes or in other words, there are several columns in the modes which are repeated. Consequently, we introduce the *compressed* s-mode, denoted by H_s in the following manner:

$$H_s = A_s J_s \tag{8}$$

where J_s is a linear transformation which associates a *redundant block-Hankel* matrix A_s to a *non-redundant Hankel* matrix H_s. For instance, we give the following example for the $4 \times 5 \times 3$ 1-mode,

$$\left(\begin{array}{ccccc|ccccc|ccccc} 0 & 1 & 2 & 3 & 4 & \underline{1} & \underline{2} & \underline{3} & \underline{4} & 5 & \underline{2} & \underline{3} & \underline{4} & \underline{5} & 6 \\ 1 & 2 & 3 & 4 & 5 & \underline{2} & \underline{3} & \underline{4} & \underline{5} & 6 & \underline{3} & \underline{4} & \underline{5} & \underline{6} & 7 \\ 2 & 3 & 4 & 5 & 6 & \underline{3} & \underline{4} & \underline{5} & \underline{6} & 7 & \underline{4} & \underline{5} & \underline{6} & \underline{7} & 8 \\ 3 & 4 & 5 & 6 & 7 & \underline{4} & \underline{5} & \underline{6} & \underline{7} & 8 & \underline{5} & \underline{6} & \underline{7} & \underline{8} & 9 \end{array}\right) \tag{9}$$

where for simplicity we have denoted $x[n]$ by its index n. In addition, we have underlined the common columns. Consequently, the matrix J_s is determined so as to produce the following "compressed" 1-mode:

$$\left(\begin{array}{ccccc|c|c} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{array}\right). \tag{10}$$

Remark the Hankel structure of matrix (10). If we note $\mathtt{nc}(.)$ the number of columns, it comes:

$$\mathtt{nc}(H_s) = \sum_{s' \neq s} I_{s'} - 1 < \mathtt{nc}(A_s) = \prod_{s' \neq s} I_{s'} \tag{11}$$

and thus H_s has less columns than A_s. Decreasing the column dimension of the mode modifies the dominant singular space, denoted by $\mathtt{R}(.)$. In other words, we have $\mathtt{R}(H_s) \neq \mathtt{R}(A_s)$. So, for a given non-deficient matrix, D_s, we have to satisfy:

$$\mathtt{R}(H_s D_s) = \mathtt{R}(A_s). \tag{12}$$

Toward this end, consider the correlation matrix of the s-mode: $C^{(s)} = A_s A_s^H$. For the 1-mode case, we have:

$$\begin{aligned} [C^{(1)}]_{k,l} &= \sum_{i_2=0}^{I_2-1} \sum_{i_3=0}^{I_3-1} [\mathcal{H}]_{k,i_2,i_3} [\mathcal{H}^*]_{l,i_2,i_3} \\ &= \sum_{i_2=0}^{I_2-1} \sum_{i_3=0}^{I_3-1} x[k+i_2+i_3] x^*[l+i_2+i_3]. \end{aligned}$$

It can be noticed that some terms are redundant in this last equation. In order to remove this redundancy, we obtain after some derivations:

$$[C^{(1)}]_{k,l} = \sum_{i=0}^{I_2+I_3-2} d_{(I_2,I_3)}[i+1] x[k+i] x^*[l+i]$$

where:

$$d_{(I_2,I_3)}[i] = \begin{cases} i & \text{if } 1 \leq i < \min(I_2,I_3) \\ \min(I_2,I_3) & \text{if } \min(I_2,I_3) \leq i < \max(I_2,I_3) \\ I_2+I_3-i & \text{if } \max(I_2,I_3) \leq i < I_2+I_3 \\ 0 & \text{elsewise.} \end{cases}$$

The weighting function $i \mapsto d_{(I_2,I_3)}[i]$ is plotted in figure 1.

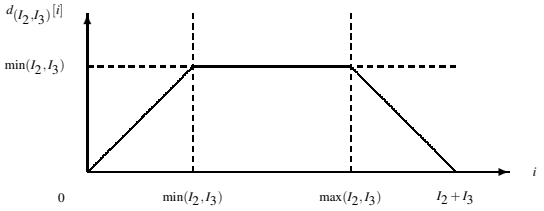


Fig. 1. Weighting function $d_{(I_2,I_3)}[i]$

It represents the number of times each column of the 1-mode is repeated. For instance, in the example of matrix (9), the weights are $\{1,2,3,3,3,2,1\}$.

The correlation matrix of the 1-mode can then be rewritten as a function of the compressed 1-mode H_1:

$$[C^{(1)}]_{k,l} = \sum_{i=0}^{N-I_1-1} d_{(I_2,I_3)}[i+1] [H_1]_{k,i} [H_1^*]_{l,i}.$$

Then define the $(N-I_1) \times (N-I_1)$ diagonal matrix D_1,whose diagonal coefficients are $[D_1]_{(i,i)} = \sqrt{d_{(I_1,I_2)}[i]}$. We obtain:

$$C^{(1)} = H_1 D_1^2 H_1^H. \tag{13}$$

Consequently, the left singular vectors of matrix A_1 can be obtained as the left singular vectors of matrix $H_1 D_1$ and thus $\mathtt{R}(H_1 D_1) = \mathtt{R}(A_1)$. In a similar way, we can build D_2 and D_3 such as $\mathtt{R}(H_2 D_2) = \mathtt{R}(A_2)$ and $\mathtt{R}(H_3 D_3) = \mathtt{R}(A_3)$. Obviously, we have:

$$\mathtt{nc}(H_s D_s) = \mathtt{nc}(H_s) < \mathtt{nc}(A_s). \tag{14}$$

Thus the SVD of $H_s D_s$ can be computed much faster than that of A_s. The fast algorithm for computing the full or truncated HOSVD of tensor $\mathcal{H}$ is summarized in table 2.

Table 2. Fast HOSVD algorithm

Operation	Full cost	Truncated cost
SVD of $H_1 D_1$	$k_{\mathrm{f}} I_1^2 (I_2+I_3)$	$k_{\mathrm{t}} r_1 I_1 (I_2+I_3)$
SVD of $H_2 D_2$	$k_{\mathrm{f}} I_2^2 (I_1+I_3)$	$k_{\mathrm{t}} r_2 I_2 (I_1+I_3)$
SVD of $H_3 D_3$	$k_{\mathrm{f}} I_3^2 (I_1+I_2)$	$k_{\mathrm{t}} r_3 I_3 (I_1+I_2)$
Tucker product	$6IQ$	$6rQ$

It can be noticed that the most computationally demanding step is now the Tucker product[1]. If tensor $\mathcal{S}$ does not have to be computed, then the complexity of the full HOSVD fast algorithm is always lower than $6k_{\mathrm{f}} I^3$, and that of the truncated HOSVD fast algorithm is lower than $6k_{\mathrm{t}} r I^2$. So, the compression and weighting of the modes allow a reduction of the complexity of one order of magnitude with respect to the basic HOSVD and the r-truncated HOSVD.

4.3. Implementation with Fast Fourier Transforms

To further reduce the complexity, we use the fact that the product between a Hankel matrix and a vector can be efficiently computed by means of Fast Fourier Transforms (FFT) [4, pp. 201–202].

By introducing those fast products into the orthogonal iteration method, the cost of the full SVD of a $n \times l$ matrix with $n < l$ and $N = n+l-1$ is reduced to $k_1 nN \log_2(N) + k_2 n^3$, where the constant k_1 is related to the Hankel matrix / vector products, and the constant k_2 is related to the QR factorizations (see [4, pp. 410–411] for more details). In the same way, the cost of a rank-r truncated SVD is $k_1 rN \log_2(N) + k_2 nr^2$.

The ultra-fast algorithm for computing the full or truncated HOSVD of tensor $\mathcal{H}$ is summarized in table 3. Again, it can be noticed that the most computationally demanding step is the Tucker product[2]. If tensor $\mathcal{S}$ does not have to be computed, then the complexity of the full HOSVD fast algorithm is always lower than $9k_1 I^2 \log_2 I + 3k_2 I^3$, and that of the truncated HOSVD fast algorithm is lower than $9k_1 rI \log_2 I + 3k_2 r^2 I$. Therefore we achieve one more order of magnitude of complexity reduction.

[1] In fact, the redundancy in $\mathcal{H}$ implies that the Tucker product involves several times the same matrix/vector products. Therefore it can be computed faster than in the non-structured case. However this does not lead to a significant reduction of the complexity.

[2] The redundancy in $\mathcal{H}$ also implies that the Tucker product involves several times the same Hankel matrix / vector products. Therefore it can be computed even faster than in the fast algorithm. Unfortunately, this does not lead to a significant reduction of the complexity either.

Table 3. Ultra-fast HOSVD algorithm

Operation	Full cost	Truncated cost
SVD of $\mathtt{H}_1 D_1$	$3k_1 I_1 I \log_2 I + k_2 I_1^3$	$3k_1 r_1 I \log_2 I + k_2 r_1^2 I_1$
SVD of $\mathtt{H}_2 D_2$	$3k_1 I_2 I \log_2 I + k_2 I_2^3$	$3k_1 r_2 I \log_2 I + k_2 r_2^2 I_2$
SVD of $\mathtt{H}_3 D_3$	$3k_1 I_3 I \log_2 I + k_2 I_3^3$	$3k_1 r_3 I \log_2 I + k_2 r_3^2 I_3$
Tucker product	$6IQ$	$6rQ$

5. ADAPTIVE SIGNAL PROCESSING

In this section, tensor $\mathcal{H}(t)$ is supposed to have slow time variations, and our objective is to efficiently update its HOSVD.

A very classical approach for tracking the SVD of a time-varying matrix consists in interlacing the update of the data with one or a few steps of a standard SVD algorithm, such as the orthogonal iteration method [10]. Following this idea, an efficient way of updating $U^{(s)}(t)$ consists in replacing the exact SVD of $A_s(t)$ in table 1 by one step of the *orthogonal iteration* method:

- $Z^{(s)}(t) = A_s(t) A_s(t)^H U^{(s)}(t-1)$
- $U^{(s)}(t) R^{(s)}(t) = Z^{(s)}(t)$.

The first operation is a matrix product; the second one is a QR-factorization. In the case of the full HOSVD, the QR factorization can be computed by means of the Householder QR method [4, pp. 224–225], or the Fast Givens QR method [4, pp. 228–229]. Its cost is $\frac{4}{3} I_s^3$ flops. In the case of the truncated HOSVD, the fastest approach is the Fast Givens QR method, whose cost is $2r_s^2 I_s$ flops. The overall cost of the sequential HOSVD algorithm obtained in this way is of the same order as that of the basic HOSVD algorithm, with a smaller multiplicative constant.

According to table 4, we can say that for square tensors the computational cost of the basic HOSVD is $O(I^4)$ and that of the r-truncated version is $O(rI^3)$, whereas the ultra-fast truncated HOSVD derived in the same way has a final complexity of $O(rI \log_2(I))$ (since we consider that $270 \log_2(I) \gg 6r$).

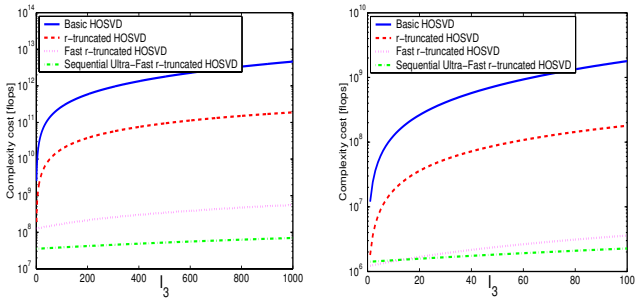
Table 4. Complexities of the sequential HO-SVD algorithms

Algorithm	Full cost	Truncated cost
Basic HO-SVD	$6I^4$	$12rI^3$
Fast HO-SVD	$16I^3$	$24rI^2 + 6r^2 I$
Ultra-fast HO-SVD	$270I^2 \log_2 I + 4I^3$	$270rI \log_2 I + 6r^2 I$

6. NUMERICAL SIMULATIONS

In this part, we represent on figure 2 the complexity of the full Basic HOSVD, the r-truncated HOSVD, the fast (with compressed and windowed modes) r-truncated HOSVD and the sequential ultra-fast r-truncated HOSVD. The methodology is the following: we fix the two first dimensions of the tensor and we vary the last one.

As we can see a large complexity gain can be achieved. We note that the most important part of the complexity gain is obtained through the SVD of the compressed-windowed Hankel-structured modes. In addition, the adaptive/sequential approach is very attractive for tensors of very large dimensions.



(a) (b)

Fig. 2. Comparison of the complexities, (a) $499 \times 699 \times i_3$ tensor where $i_3 \in [0:999]$ and $r = 30$ and (b) $99 \times 99 \times i_3$ tensor where $i_3 \in [0:99]$ and $r = 10$.

7. CONCLUSIONS

In this paper, we decreased the computational cost of the r-truncated HOSVD of structured tensors. Our solution is based on the fact that the HOSVD can be reduced to the SVD of three non-redundant (all columns are different) Hankel matrices whose columns are multiplied by a given weighting function. This operation is a first step which allows the gain of one order of magnitude. To further reduce the complexity, we efficiently compute the products between Hankel matrices and vectors by means of Fast Fourier Transforms. Finally, in an adaptive context, we propose a sequential implementation of the SVD by means of the orthogonal iteration algorithm. Our fastest implementation of the HOSVD has a complexity of $O(rI \log_2(I))$ in the case of square $(I \times I \times I)$ tensors.

8. REFERENCES

[1] P. Comon, "Tensor Decomposition, state of the art and applications", *IMA Conf. Mathematics in Signal Processing*, 2000.

[2] N.D. Sidiropoulos, "Low-Rank Decomposition of Multi-Way Arrays: A Signal Processing Perspective", *IEEE Workshop on Sensor Array and Multichannel processing (SAM2004)*, July 18-21, 2004

[3] J.M. Papy, L. De Lathauwer, S. Van Huffel, "Exponential data fitting using multilinear algebra. The single-channel and the multichannel case", *Numerical Linear Algebra and Applications*, vol. 12, no. 8, Oct. 2005, pp. 809-826.

[4] G.H. Golub, C.F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, Maryland: Johns Hopkins University Press, 1996.

[5] L.R. Tucker, "Some Mathematical Notes on Three-mode Factor Analysis", *Psychometrika*, Vol. 31, 1966, pp. 279-311.

[6] R. A. Harshman and M. E. Lundy, "PARAFAC: Parallel factor analysis", *Computational Statistics and Data Analysis*, 18, 39-72, 1994.

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

[8] L. De Lathauwer. *Signal Processing based on Multilinear Algebra*, PhD thesis, Katholieke Universiteit Leuven, 1997.

[9] T. D. Howell, "Global properties of tensor rank", *Linear Algebra and Applications*, vol 22, pp 9-23, 1978.

[10] R. Badeau, G. Richard and B. David, "Sliding window adaptive SVD algorithms", *IEEE Trans. on Signal Processing*, vol. 52, no. 1, pp. 1-10, janvier 2004.

[11] H. Wang, N. Ahuja, "Facial Expression Decomposition", *International Conference on Computer Vision (ICCV)*, 2003.

[12] B. Savas, *Analyses and Tests of Handwritten Digit Recognition Algorithms*, Master's Thesis, Department of Mathematics, Linkoping University, 2003.

[13] J.-T. Sun, "CubeSVD: A Novel Approach to Personalized Web Search", *International World Wide Web Conference*, May 10-14, 2005, Japan.