ADAPTIVE MULTILINEAR SVD FOR STRUCTURED TENSORS
Remy Boyer, Roland Badeau

To cite this version:
Remy Boyer, Roland Badeau. ADAPTIVE MULTILINEAR SVD FOR STRUCTURED TENSORS. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP’06), 2006, Toulouse, France. hal-00577274

HAL Id: hal-00577274
https://hal.archives-ouvertes.fr/hal-00577274
Submitted on 17 Mar 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution – NonCommercial – NoDerivatives| 4.0 International License
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 (i.e. arrays with more than two indexes) and plays an important role in various domains. Unfortunately, the computational cost of this decomposition is very high. In the case of structured tensors, these modes are highly redundant 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 square \((I \times I \times I)\) tensors the complexity of the basic full-HOSVD is \(O(I^4)\) and \(O(r^3)\) 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 related 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 (i.e. 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 (i.e. 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 \(O(r^3)\) 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 \(I_1 \times I_2 \times I_3\) tensor. We define the modes (also called “matrix unfoldings”) \(A_1, A_2, A_3\) as follows:

\[
\begin{align*}
[A_1]_{i_1, (i_2-1)I_3 + i_2} &= [A]_{i_1, i_2 i_3}, \\
[A_2]_{i_2, (i_1-1)I_3 + i_1} &= [A]_{i_2, i_1 i_3}, \\
[A_3]_{i_3, (i_1-1)I_2 + i_1} &= [A]_{i_3, i_1 i_2},
\end{align*}
\]

where we have denoted the entries of \(A\) by \([A]_{i_1 i_2 i_3}\) 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 \(A\) 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 \(A\) can be written as the product:

\[
A = S \times U^{(1)} \times U^{(2)} \times U^{(3)}
\]

in which \(\times_s\) represents the Tucker \(s\)-mode product [8], \(U^{(s)}\) is an unitary \(I_s \times I_s\) matrix and \(S\) is an all-orthogonal and ordered \(I_1 \times I_2 \times I_3\) tensor. All-orthogonality means that the matrices \(S_{i_1-1, i_1-1}\) obtained by fixing the \(s\)-th index to \(i_s\) are mutually orthogonal w.r.t. the standard inner product. Ordering means that \(|\|S_{i_1-1,0}\| \geq \|S_{i_1-1,1}\| \geq \ldots \geq \|S_{i_1-1, I_1-1}\|\geq 0\) for all possible values of \(s\). The Frobenius-norms...
\[ S = A \times_1 U_1^{(1)H} \times_2 U_2^{(2)H} \times_3 U_3^{(3)H} \]  

where \((\cdot)^H\) denotes the complex conjugate.

### 2.3. Mode decompositions

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

\[
\begin{align*}
A_1 &= U_1^{(1)} S_1 \left( U_2^{(2)} \otimes U_3^{(3)} \right)^T, \\
A_2 &= U_2^{(2)} S_2 \left( U_3^{(3)} \otimes U_1^{(1)} \right)^T, \\
A_3 &= U_3^{(3)} S_3 \left( U_1^{(1)} \otimes U_2^{(2)} \right)^T,
\end{align*}
\]

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

### 3. COMPLEXITY OF FULL AND TRUNCATED HOSVD

Let \(N = I_1 + I_2 + I_3 - 1\), \(Q = I_1 J_2 J_3\) and \(I = \lceil \frac{N}{1+3} \rceil\). 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 HOSVD

The calculation of the full HOSVD of tensor \(A\) requires the computation, for all \(s \in [1:3]\), of the left factor \(U_1^{(s)}\) in the full SVD of matrix \(A_s\), as defined above. Once the three matrices \(U_1^{(s)}\) have been obtained, the tensor \(S\) can be computed by means of the Tucker product

\[
S = A \times_1 U_1^{(1)H} \times_2 U_2^{(2)H} \times_3 U_3^{(3)H}. \tag{6}
\]

According to [4, pp. 253–254], the average computational cost of the full SVD of an \(n \times l\) matrix with \(n < l\) is \(k_0n^2l\) flops, where the constant \(k_0\) 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 \(k_0\)). 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 \(k_0n^2lI_2I_3\) 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 \((3k_0 + 6)I^4\).

### 3.2. Computation of the truncated HOSVD

In many applications, we are interested in computing the HOSVD truncated at orders \((r_1, r_2, r_3)\) \((r_i\) is often supposed to be much lower than \(I_i)\). 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 \(r_i\). 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 \(k_0r^2nl\), where the constant \(k_0\) depends on the algorithm (\(k_0\) is generally greater than \(k_0\)). Therefore, applied to matrix \(A_s\), its cost is \(O(r^3)\) flops for square tensors. Note that we assume \((1) I_0 \gg \log_2(I_0)\) and \(2 \log_2(I_0)\) is the same order as \(r_0\). The computational costs of the \(r\)-truncated HOSVD are summarized in Table 1.

| Table 1. Full and \(r\)-truncated HOSVD Algorithms |

<table>
<thead>
<tr>
<th>Operation</th>
<th>Full</th>
<th>(r)-truncated HOSVD</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD of (A_1)</td>
<td>(k_0r_1Q)</td>
<td>(k_0r_1Q)</td>
</tr>
<tr>
<td>SVD of (A_2)</td>
<td>(k_0r_2Q)</td>
<td>(k_0r_2Q)</td>
</tr>
<tr>
<td>SVD of (A_3)</td>
<td>(k_0r_3Q)</td>
<td>(k_0r_3Q)</td>
</tr>
<tr>
<td>Tucker product</td>
<td>(6Q)</td>
<td>(6rQ)</td>
</tr>
<tr>
<td>Total</td>
<td>((3k_0 + 6)I_0Q)</td>
<td>((3k_0 + 6)r_0Q)</td>
</tr>
</tbody>
</table>

### 4. FAST ALGORITHMS FOR HANKEL-STRUCTURED TENSORS

This section is dedicated to Hankel-structured tensors but other structures as for instance circular 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:

\[
[A]_{i_1 i_2 i_3} = x[i_1 + i_2 + i_3] \tag{7}
\]

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 \(H_i\) in the following manner:

\[
H_i = A_s I_i \tag{8}
\]

where \(J_i\) is a linear transformation which associates a redundant block-Hankel matrix \(A_s\) to a non-redundant Hankel matrix \(H_i\). For instance, we give the following example for the \(4 \times 5 \times 3\) 1-mode:

\[
\begin{bmatrix}
0 & 1 & 2 & 3 & 4 \\
1 & 2 & 3 & 4 & 5 \\
2 & 3 & 4 & 5 & 6 \\
3 & 4 & 5 & 6 & 7
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & \frac{1}{2} & \frac{1}{2} & 1 & \frac{1}{2} \\
\frac{1}{2} & \frac{1}{2} & 1 & \frac{1}{2} & 1 \\
\frac{1}{2} & 1 & \frac{1}{2} & 1 & \frac{1}{2} \\
1 & \frac{1}{2} & \frac{1}{2} & 1 & \frac{1}{2}
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & 1 & 2 & 3 & 4 \\
1 & 2 & 3 & 4 & 5 \\
2 & 3 & 4 & 5 & 6 \\
3 & 4 & 5 & 6 & 7
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & 1 & 2 & 3 & 4 \\
1 & 2 & 3 & 4 & 5 \\
2 & 3 & 4 & 5 & 6 \\
3 & 4 & 5 & 6 & 7
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & 1 & 2 & 3 & 4 \\
1 & 2 & 3 & 4 & 5 \\
2 & 3 & 4 & 5 & 6 \\
3 & 4 & 5 & 6 & 7
\end{bmatrix}
\]
where for simplicity we have denoted \( x[n] \) by its index \( n \). In addition, we have underlined the common columns. Consequently, the matrix \( J_k \) is determined so as to produce the following “compressed” 1-mode:

\[
\begin{pmatrix}
0 & 1 & 2 & 3 & 4 & 5 & 6 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 \\
2 & 3 & 4 & 5 & 6 & 7 & 8 \\
3 & 4 & 5 & 6 & 7 & 8 & 9
\end{pmatrix}
\]  

(10)

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

\[
C^{(1)} = H_1 D_1^{1/2} H_1^*.
\]  

(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 \( R(H_1 D_1) = R(A_1) \). In a similar way, we can build \( D_2 \) and \( D_3 \) such as \((H_2 D_2) = (A_2)\) and \((H_3 D_3) = (A_3)\). Obviously, we have:

\[
\text{nc}(H_2 D_2) = \text{nc}(H_3 D_3) < \text{nc}(A_1).
\]  

(14)

Thus the SVD of \( H_2 D_2 \) can be computed much faster than that of \( A_1 \). The fast algorithm for computing the full or truncated HOSVD of tensor \( \mathcal{H} \) is summarized in table 2.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Full cost</th>
<th>Truncated cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD of ( H_1 D_1 )</td>
<td>( k_1 f(1+I_1) )</td>
<td>( k_1 r(1+I_1) )</td>
</tr>
<tr>
<td>SVD of ( H_2 D_2 )</td>
<td>( k_2 f(I_2+I_3) )</td>
<td>( k_2 r(I_2+I_3) )</td>
</tr>
<tr>
<td>SVD of ( H_3 D_3 )</td>
<td>( k_3 f(I_1+I_2) )</td>
<td>( k_3 r(I_1+I_2) )</td>
</tr>
<tr>
<td>Tucker product</td>
<td>6IQ</td>
<td>6rQ</td>
</tr>
</tbody>
</table>

It can be noticed that the most computationally demanding step is now the Tucker product\(^1\). If tensor \( S \) does not have to be computed, then the complexity of the full HOSVD fast algorithm is always lower than \( 6k_3 r^2 I \), and that of the truncated HOSVD fast algorithm is lower than \( 6k_3 r I^2 \). 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 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 \( k_1 n \log_2(N) + k_2 n^2 \), 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 \( k_1 n \log_2(N) + k_2 n r^2 \).

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 \( S \) does not have to be computed, then the complexity of the full HOSVD fast algorithm is always lower than \( 9k_3 l^2 \log_2 I + 3k_2 l^2 I \), and that of the truncated HOSVD fast algorithm is lower than \( 9k_3 r I \log_2 I + 3k_3 r^2 I \). 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

<table>
<thead>
<tr>
<th>Operation</th>
<th>Full cost</th>
<th>Truncated cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVD of $B_1 D_1$</td>
<td>$3k_1 I_1 \log J + k_1 I_1^2$</td>
<td>$3k_1 I_1 \log J + k_1 I_1^2$</td>
</tr>
<tr>
<td>SVD of $B_2 D_2$</td>
<td>$3k_1 I_2 \log J + k_2 I_2^2$</td>
<td>$3k_1 I_2 \log J + k_2 I_2^2$</td>
</tr>
<tr>
<td>SVD of $B_3 D_3$</td>
<td>$3k_1 I_3 \log J + k_3 I_3^2$</td>
<td>$3k_1 I_3 \log J + k_3 I_3^2$</td>
</tr>
<tr>
<td>Tucker product</td>
<td>$6Q$</td>
<td>$6rQ$</td>
</tr>
</tbody>
</table>

### 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 interleaving 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 $U^{(i)}(t)$ consists in replacing the exact SVD of $A_i(t)$ in table 1 by one step of the orthogonal iteration method:

- $Z^{(i)}(t) = A_i(t) A_i(t)$
- $U^{(i)}(t) R^{(i)}(t) = Z^{(i)}(t)$

The first operation is a product matrix; 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 $4t_i^3$ flops. In the case of the truncated HOSVD, the fastest approach is the Fast Givens QR method, whose cost is $2r_i^2 I_i$ 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 $O(I^3)$ and that of the $r$-truncated version is $O(I^3)$, whereas the ultra-fast truncated HOSVD derived in the same way has a final complexity of $O(r I \log J)$ (since we consider that $270 \log_2(I) \gg 6r$).

### Table 4. Complexities of the sequential HO-SVD algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Full cost</th>
<th>Truncated cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic HO-SVD</td>
<td>$6J^2$</td>
<td>$12J^2$</td>
</tr>
<tr>
<td>Fast HO-SVD</td>
<td>$16f^3$</td>
<td>$24r I^2 + 6r I^3$</td>
</tr>
<tr>
<td>Ultra-fast HO-SVD</td>
<td>$270 J^2 \log J + 4f^3$</td>
<td>$270 r I \log J + 6r I^3$</td>
</tr>
</tbody>
</table>

### 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.

### 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 $O(r I \log J)$ in the case of square $(I \times I \times J)$ tensors.

### 8. REFERENCES