doi:10.2498/iti.2012.0466

A Framework for Data Representation, Processing, and Dimensionality Reduction with the Best-Rank Tensor Decomposition

Bogusław Cyganek
AGH University of Science and Technology, Al. Mickiewicza 30, Kraków 30-059, Poland
E-mail: cyganek@agh.edu.pl

Abstract. *The paper addresses the problem of efficient multi-dimensional data representation, processing and dimensionality reduction. For this purpose the framework for the best rank-R tensor decomposition is presented. This allows any multi-dimensional data reduction in accordance with chosen ranks. Since computations of tensor decomposition require floating-point operations, we propose special data scaling procedure to allow memory efficient representation in the fixed-point representation. The proposed method is exemplified with processing of the monochrome and color video sequences. The method shows promising results and can be easily applied to other types of multi-dimensional data.*

Keywords. Tensor best rank-*R* decomposition, dimensionality reduction, data mining.

1. Introduction

The newest information technologies involve generation and processing of huge amounts of data. The problem of efficient information storage, search, and retrieval are not only concerned about newest telecommunication and computer technologies but also they require new methods of data representation and processing. In this paper we address these problems for video content data and present a framework for multi-dimensional data representation, processing, and dimensionality reduction based on the best-rank tensor decomposition.

The well known method of data dimensionality reduction is the Principal Component Analysis (PCA) [6]. However, in the case of multi-dimensional data, such as seismic, video, hyper-spectral imaging, etc. much better results were shown with the tensor based approach in which the input data is treated as a multi-dimensional 'cube' of values. However, the real benefit of such representation can be appreciated realizing that thanks to this method the dominating dimensions can be extracted which most of all influence information content. For this purpose the three most common methods are: the Higher-Order Singular Value Decomposition (HOSVD), best rank-1 and rank-(R_1, R_2, …, R_P) approximations [8][9]. The so called truncated HOSVD results in excessive errors and therefore can be treated only as a crude approximation or can serve as an initialization method. In this respect better results can be obtained with the best rank-1 decomposition [12]. However, as was recently shown by Wang *et al.*, the rank-(R_1, R_2, …, R_P) approximation, although computationally more demanding, can lead to the superior results in respect to dimensionality reduction and reconstruction error [13].

The main contribution of this paper is presentation of the efficient parallel implementation of the rank-(R_1, R_2, …, R_P) tensor approximation in applications to dimensionality reduction in video information. This can be used to information retrieval, object recognition and for data compression. However, since the computations involve floating point arithmetic this shows inefficient for huge data repositories. Therefore, our next contribution is a proposition of a method for efficient tensor scaling and processing with the fixed point format. This allows similar computational accuracy with greatly lower memory needs.

The rest of the paper is organized as follows. In section 2 the rank-(R_1, R_2, …, R_P) decomposition is presented. Sections 3 and 4 are devoted to the algorithmic realizations of the decomposition methods, as well as proposed scaling method, respectively. Section 5 presents experimental results. The paper ends with conclusions in section 6.

2. Rank-R Decomposition of Tensors

A recommended readings for a short introduction to tensors are [8][3][4]. The problem of a best rank-(R_1, R_2, …, R_P) approximation can be stated in an analogous way as the best rank-one approximation, as follows: Given a tensor $\mathcal{T} \in \Re^{N_1 \times N_2 \times \ldots \times N_P}$ find a tensor $\tilde{\mathcal{T}}$ having $rank_1\left(\tilde{\mathcal{T}}\right) = R_1$, $rank_2\left(\tilde{\mathcal{T}}\right) = R_2$, …, $rank_P\left(\tilde{\mathcal{T}}\right) = R_P$, that is as close as possible to the input tensor $\mathcal{T}$. This task can be formulated as minimization of the following least-squares cost function

$$\Theta\left(\tilde{\mathcal{T}}\right) = \left\|\tilde{\mathcal{T}} - \mathcal{T}\right\|_F^2, \tag{1}$$

in the sense of the Frobenius norm. Approximated this way tensor $\tilde{\mathcal{T}}$ conveys as much of the "energy", in the sense of the squared entries of a tensor, as the original tensor $\mathcal{T}$ given the rank constraints.

It can be easily observed that the assumed rank conditions mean that the approximation tensor $\tilde{\mathcal{T}}$ can be decomposed as

$$\tilde{\mathcal{T}} = \mathcal{Z} \times_1 \mathbf{S}_1 \times_2 \mathbf{S}_2 \ldots \times_P \mathbf{S}_P, \tag{2}$$

where each of the matrices $\mathbf{S}_1 \in \Re^{N_1 \times R_1}$, $\mathbf{S}_2 \in \Re^{N_2 \times R_2}$, …, $\mathbf{S}_P \in \Re^{N_P \times R_P}$ has orthonormal columns (each time, number of columns for $\mathbf{S}_k$ is given by R_k), while the tensor $\mathcal{Z} \in \Re^{R_1 \times R_2 \times \ldots \times R_P}$ has dimensions R_1, R_2, …, R_P.

It can be shown that the tensor $\mathcal{Z}$ can be computed from the original tensor $\mathcal{T}$ as

$$\mathcal{Z} = \mathcal{T} \times_1 \mathbf{S}_1^T \times_2 \mathbf{S}_2^T \ldots \times_P \mathbf{S}_P^T. \tag{3}$$

From the above one can conclude that to find the best rank-(R_1, R_2, …, R_P) approximation of $\mathcal{T}$ it is sufficient to determine only a set of $\mathbf{S}_k$ in (2), and then $\mathcal{Z}$ can be computed from (3). This leads to the following theorem [9]: Minimization of $\Xi\left(\tilde{\mathcal{T}}\right)$ in (1) is equivalent to the maximization of the following function

$$\begin{aligned}\Psi\left(\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_P\right) &= \left\|\mathcal{Z}\right\|^2 \\ &= \left\|\mathcal{T} \times_1 \mathbf{S}_1^T \times_2 \mathbf{S}_2^T \ldots \times_P \mathbf{S}_P^T\right\|^2\end{aligned} \tag{4}$$

over the matrices $\mathbf{S}_1$, $\mathbf{S}_2$, …, $\mathbf{S}_P$ which have orthonormal columns. The two functions Ψ in (4) and Θ in (1) are related as follows

$$\Theta\left(\tilde{\mathcal{T}}\right) = \left\|\mathcal{T}\right\|^2 - \left\|\Psi\right\|^2. \tag{5}$$

The above theorem was used to develop a numerical Alternating Least-Squares (ALS) method for computation of the best rank-(R_1, R_2, …, R_P) approximation of tensors, as proposed by Lathauwer *et al.* [9]. The ALS approach assumes that in each step only one of the matrices $\mathbf{S}_k$ is optimized, whereas other are kept fixed. In other words, the idea is to represent the function (4) as a quadratic expression in the components of the unknown matrix $\mathbf{S}_k$ with orthogonal columns with other matrices kept fixed, i.e.

$$\begin{aligned}&\max_{\mathbf{S}_k}\left\{\Psi\left(\mathbf{S}_k\right)\right\} = \\ &\max_{\mathbf{S}_k}\left\|\mathcal{T} \times_1 \mathbf{S}_1^T \times_2 \mathbf{S}_2^T \ldots \times_P \mathbf{S}_P^T\right\|^2.\end{aligned} \tag{6}$$

Columns of $\mathbf{S}_k$ can be obtained as an orthonormal basis for the dominating subspace of the column space of the matrix in $\hat{\mathbf{S}}_k$. In a pure matrix representation the latter can be expressed as follows (assuming the forward cyclic mode)

$$\begin{aligned}\hat{\mathbf{S}}_k = \mathbf{T}_{(k)}\big[&\mathbf{S}_{k-1} \otimes \mathbf{S}_{k-2} \otimes \ldots \\ &\otimes \mathbf{S}_1 \otimes \mathbf{S}_P \otimes \mathbf{S}_{P-1} \otimes \ldots \otimes \mathbf{S}_{k+1}\big].\end{aligned} \tag{7}$$

As already mentioned, in each step only one matrix $\mathbf{S}_k$ is computed, while other are kept fixed. Such computations are repeated until the stopping condition expressed by (5) is met or a maximal number of iterations is reached. The procedure is called the Higher-Order Orthogonal Iteration (HOOI) [9].

Dimensionality reduction and data compression require measurements of the accuracy of the approximations [10]. In image compression community the common measures of comparison are the mean-square error (MSE)

$$\begin{aligned}&MSE\left(\tilde{\mathcal{T}}, \mathcal{T}\right) = \\ &\sum_{n_1=1}^{N_1} \cdots \sum_{n_P=1}^{N_P}\Big[\tilde{\mathcal{T}}\left(n_1, n_2, \ldots, n_P\right) \\ &-\mathcal{T}\left(n_1, n_2, \ldots, n_P\right)\Big]^2\end{aligned} \tag{8}$$

and the peak signal to noise ratio (PSNR). The latter is expressed in dB, as follows [4]

$$\begin{aligned}&PSNR\left(\tilde{\mathcal{T}}, \mathcal{T}\right) = \\ &10\log_{10}\left(\frac{\mathrm{m}^2}{MSE\left(\tilde{\mathcal{T}}, \mathcal{T}\right)}\right),\end{aligned} \tag{9}$$

where *m* denotes a maximum value of a pixel (e.g. *m=255* for 8 bit images).

3. Computation of the Best-Rank Tensor Decomposition

A computation method of the best rank-(R_1, R_2, …, R_P) approximation of tensors is presented in Algorithm 3.1.

```
1. Choose a number of ranks $R_1, R_2, \ldots, R_P$, threshold $\varepsilon_{ACCEPT}$ and the max iterations $t_{max}$.

2. Initialize:  $\mathbf{S}_k^{(0)} \in \Re^{N_k \times R_k}$ for $1 \le k \le P$
   $t = 0$

3. Do:

  4. For each $k$, such that $1 \le k \le P$, compute:

    $\hat{\mathbf{S}}_k^{(t+1)} = \mathbf{T}_{(k)}\left[\mathbf{S}_{k-1}^{(t)} \otimes \mathbf{S}_{k-2}^{(t)} \otimes \ldots \otimes \mathbf{S}_1^{(t)} \otimes \mathbf{S}_P^{(t+1)} \otimes \ldots \otimes \mathbf{S}_{k+1}^{(t+1)}\right]$   (10)

    $\mathbf{S}_k^{(t+1)} = svds\left(\hat{\mathbf{S}}_k^{(t+1)}, R_k\right)$   (11)

  $\mathcal{Z}_{t+1} = \mathcal{T} \times_1 \mathbf{S}_1^{(t+1)^T} \times_2 \mathbf{S}_2^{(t+1)^T} \ldots \times_P \mathbf{S}_P^{(t+1)^T}$   (12)

  $\varepsilon = \left| \left\| \mathcal{Z}_{t+1} \right\|^2 - \left\| \mathcal{Z}_t \right\|^2 \right|$   (13)

  $t = t + 1$   (14)

while ( $\varepsilon > \varepsilon_{ACCEPT}$ ) and ( $t < t_{max}$ )

4. Store the computed $\mathcal{Z}$ and matrices $\mathbf{S}_k$.
```

Algorithm 3.1. Best rank-(R$_1$, R$_2$, …, R$_P$) decomposition of tensors

In Algorithm 3.1 the function *svds*($\mathbf{S}$,R) returns the R left leading singular vectors of the matrix $\mathbf{S}$, i.e. the ones corresponding to the R largest singular values of $\mathbf{S}$. These vectors are orthogonal. Moreover, it is a frequent case that the matrix $\hat{\mathbf{S}}_k$ in (11) has much more columns c than rows r. In such a case we can compute *svds* from the product $\hat{\mathbf{S}}_k \hat{\mathbf{S}}_k^T$, instead of the $\hat{\mathbf{S}}_k$, taking advantage of the fact that if a matrix $\mathbf{M}$=$\mathbf{SVD}^T$, then $\mathbf{MM}^T$=$\mathbf{SV}^2\mathbf{S}^T$. We use this fact also in our software realization of the Algorithm 3.1, after computing some heuristic assessment whether the conditions of prevailing number of columns, i.e. $c \gg r$, as well as substantial amount of data, i.e. $c \cdot r$, are fulfilled.

Regarding the initialization step (2) in Algorithm 3.1, in some publications these are proposed to be obtained from the HOSVD [9]. Although such strategy does not guarantee the optimal solution, in practice usually it leads to good results. However, as already mentioned, HOSVD is computationally demanding, so for larger problems Wang *et al.* propose to initialize $\mathbf{S}_k$, either with constant values, or with the uniformly distributed random numbers. These strategies when applied to image processing tasks gave almost the same results as initialization with the HOSVD [9]. Such an initialization method is also recommended in the paper by Chen *et al.* [1]. In our implementation we also follow this way and initialize $\mathbf{S}_k$ with uniform random generator.

Wang *et al.* propose rank-R decomposition, where R=R_1=R_2=…=R_P, for dimensionality reduction for any kind of multidimensional data, such as video sequences or volume data, which they call Datum-as-Is [13]. As they show, the rank-R decomposition allows exploitation of redundancies among data in all-dimensions. Thus, it implicitly encodes the covariances among data. In effect this leads to much higher compression ratio and better data representation for object classification. For special case of 3D data, such as video, they propose another rank-R tensor approximating algorithm which is based on slice projection of 3D tensors, and which allows faster convergence.

To appreciate compression properties of the best rank-(R_1, R_2, …, R_P) decomposition of tensors let us compare number of memory required for the original data tensor $\mathcal{T} \in \Re^{N_1 \times N_2 \times \ldots \times N_P}$, as compared to its approximated version (2), in which $\mathbf{S}_k \in \Re^{N_k \times R_k}$ and $\mathcal{Z} \in \Re^{R_1 \times R_2 \times \ldots \times R_P}$. Storage of $\mathcal{T}$ requires

$$D_0 = N_1 N_2 \ldots N_P . \tag{15}$$

On the other hand $\tilde{\mathcal{T}}$ requires

$$D_1 = R_1 R_2 \ldots R_P + \sum_{k=1}^{P} N_k R_k . \tag{16}$$

If we assume small values of R_k as compared to N_k, which is a frequent case, then the difference between D_0 and D_1 gets obvious, i.e. $D_1 \ll D_0$. For instance, for a 128 long sequence of monochrome video of resolution 640×480 D_0=37.5 MBytes. Assuming that $R_1=R_2=\ldots=R_P=48$ we obtain D_1= 0,163 MBytes. Hence, the compression ratio in this case would be 230:1. However, when considering storage of $\mathbf{S}_k$ and $\mathcal{Z}$ we have to account for the dynamical range of their data which usually require more than a byte per element.

4. Implementation Issues

In general the computed matrices $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{S}_3$, and the core tensor $\mathcal{Z}$, can contain values of high arithmetical dynamics, i.e. these can be positive or negative, and also can have large exponent. Thus, in general case we need to store floating point values. Depending on a computer system they usually require up to 8 bytes each (in C++ they are denoted with the *double* data type). However, in many cases it is possible to use more efficient representation with the fixed point. Unfortunately, that one is not a built-in type in C++, although it is quite straightforward to create a proper class to achieve this goal, as for instance the *TFixedFor* template class from the attached software framework [5][4]. The tensor decomposition classes in our framework are template ones which makes them flexible to use any data type, such as the mentioned *TFixedFor*. Thanks to this, when processing monochrome images we can limit ourselves to 2 bytes per data, stored as 16 bits integer and 16 bit of fractional. This gives us the precision of $2^{-16}\approx$15e-6 and the dynamic range of $\pm 2^{15}$ which more frequently than not is sufficient for majority of image processing routines. Thanks to this simple technique we can reach 4 times memory reduction as compared to the most obvious double data representation. This is a substantial gain when processing tensors, since many algorithms require their representation in memory as one entity.

In the case of the presented in this section data compression we can go even further with savings on data representation. For this purpose let us rescale each tensor in equation (2), as follows

$$\tilde{\mathcal{T}} = \lambda \lfloor \mathcal{Z} \rfloor \times_1 \lfloor \mathbf{S}_1 \rfloor \times_2 \lfloor \mathbf{S}_2 \rfloor \times_3 \lfloor \mathbf{S}_3 \rfloor \qquad (17)$$

where $\lfloor \mathbf{S}_i \rfloor$ denotes a scaling values of a matrix $\mathbf{S}_i$ to a certain range r, and λ denotes a scaling parameter. The scaling of the matrices can be achieved in the following simple steps:

1. Find the maximal absolute value s_{max} of the matrix $\mathbf{S}_i$.
2. Each element multiply by r/s_{max} (assuming $s_{max}\neq 0$).

The same procedure is applied to the core tensor to find out a value of z_{max}. In result the additional scalar

$$\lambda = \frac{z_{\max}}{r^{P+1}} \prod_{i=1}^{P} s_{\max}^{(i)} \qquad (18)$$

is introduced which needs to be stored. However, this cost is negligible. Thanks to this, each element of the video tensors in (2) can be stored on only one or two bytes. This inevitably adds some quantization errors. However, as shown by our experiments, this adds only small error to the final reconstruction (this is in the range of 0-3 values of worse PSNR). Results shown in Table 1 relate to the aforementioned procedure. Finally, yet another (but slightly heuristic) observation is that for many datasets the scaled core tensor $\lfloor \mathcal{Z} \rfloor$ in (17) usually is sparse which opens further possibility of reductions in data storage. This feature was not exploited in the presented experiments, however.

5. Experimental Results

The best rank-(R_1, R_2, …, R_P) decomposition of tensors can be used to approximate any data tensor. This feature can be used to compress video sequences, as will be shown in this short example. The input video sequence will be represented as a 3D or 4D tensor $\mathcal{T}$ for monochrome and color sequences, respectively. This after the best rank decomposition is represented exclusively by the three (or four) matrices $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{S}_3$, ($\mathbf{S}_4$) and the core tensor $\mathcal{Z}$, as expressed in equation (2). Presented hereafter decompositions were computed with the C++ implementation of the Algorithm 3.1 in the software framework from [5][4]. Results for the *Hamburg Taxi* sequence (Made publicly available by the group of Prof. Dr. H.-H. Nagel from the Fakultät für Informatik of the Karlsruhe Institute of Technology [7]), for different values of the ranks R_1, R_2, R_3, are presented in next figure.

The original sequence of 'Hamburg Taxi' counts 41 monochrome frames, each of resolution 256×191. Selected 5 frames are shown in the Figure 1 (middle column). This requires at least D_0= 2 004 736 bytes in a non-reduced representation.

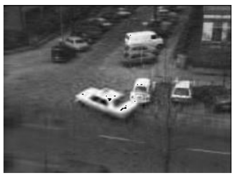
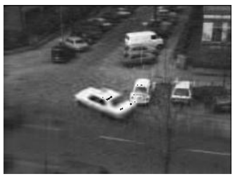
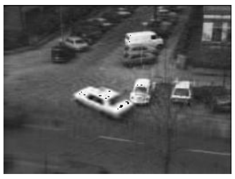
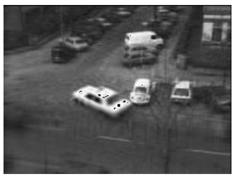
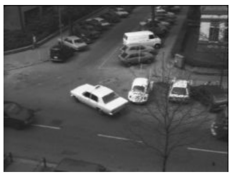
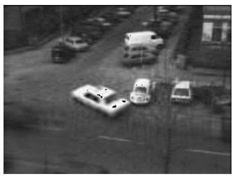
Frame No	Original sequence 256×191×41	Rank-40-40-15 compressed
1		
3		
5		
7		
9		

Figure 1. Selected frames from the 'Hamburg taxi' sequence (middle). Sequence in the rank-40-40-15 representation (right)

The first compressed version, which is shown in Figure 1 (right column), was obtained for the ranks R_1=40, R_2=40, and R_3=15, respectively. In accordance with (16) this requires D_1=42 495 elements to be stored. In this case the compression results and the accuracies are satisfactory.

The second compressed version of 'Hamburg Taxi' was obtained for the ranks R_1=30, R_2=20, and R_3=10. In this case the storage parameter D_1=17 910. Storage of the same sequence in JPEG format requires 473 503 bytes. Initialization of the matrices in Algorithm 3.1 was done with the random generator with uniform distribution. The stop criterion ε_{ACCEPT} was set to 0.05 to speed up computations.

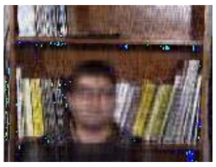
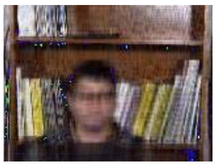
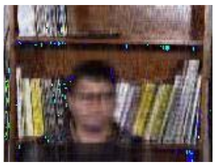
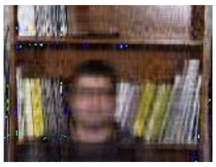
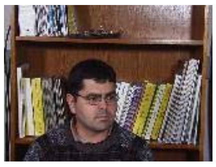
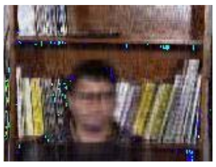
Frame No	Original color sequence 160×120	Rank-20-20-3-3 compressed
1		
2		
3		
4		
5		

Figure 2. Input frames from the 'Georgia tech face' sequence (middle). This sequence after the rank-20-20-3-3 decomposition (right)

Table 1 shows the achieved compression ratios and reconstruction errors for the two compressed sequences of 'Hamburg Taxi' shown in Figure 1.

Table 1. Achieved compression ratios and reconstruction errors for the seqence 'Hamburg Taxi' for different ranks

Ranks	Compression ratio	MSE	PSNR [dB]
40-40-15	48:1	40.3	32.1
30-20-10	112:1	83.6	28.9

Next group of experiments was devoted to the color video sequences, which naturally form 4D tensors. Figure 2 depicts few frames from the Georgia tech face, available at [7]. This sequence after the rank-20-20-3-3 decomposition is presented in Figure 2 (right column). Achieved compression ratios and reconstruction errors for this sequence and ranks 10-20-3-3 are presented in Table 2.

Table 2. Achieved compression ratios and reconstruction errors for the sequence from Figure 2.

Ranks	Compression ratio	MSE	PSNR [dB]
20-20-3-3	16 : 1	146.87	27.1
10-20-3-3	24 : 1	283.6	26.7

In this case the visual inaccuracies are significant due to somehow arbitrarily chosen ranks.

6. Conclusions

In the paper a framework for data representation, processing, and dimensionality reduction based on the best-rank tensor decomposition is presented. An efficient parallel implementation of the rank-(R_1, R_2, …, R_P) tensor approximation in applications to dimensionality reduction for large and multi-dimensional data sets is proposed. This can be used to information retrieval, object recognition and for data compression. A method for efficient tensor scaling and processing with the fixed point format is also presented. This allows similar computational accuracy with greatly lowered memory requirements. The method was tested in the context of data reduction in monochrome and color video sequences, although it can be used with any other types of data, such as seismic, EEG, etc.

7. Acknowledgements

The presented work was financially supported by the Polish National Science Centre NCN, contract no. DEC-2011/01/B/ST6/01994.

8. References

[1] Chen J, Saad Y. On the Tensor SVD and the Optimal Low Rank Orthogonal Approximation of Tensors. SIAM Journal on Matrix Analysis and Applications, Vol. 30, No. 4, pp. 1709-1734; 2009.

[2] Cichocki A, Zdunek R, Amari S. Nonnegative Matrix and Tensor Factorization. IEEE Signal Processing Magazine, Vol. 25, No. 1, pp. 142-145; 2008.

[3] Cichocki A, Zdunek R, Phan AH, Amari S-I. Nonnegative Matrix and Tensor Factorizations. Applications to Exploratory Multi-way Data Analysis and Blind Source Separation. Wiley; 2009.

[4] Cyganek B, Siebert JP. An Introduction to 3D Computer Vision Techniques and Algorithms, Wiley; 2009.

[5] Cyganek B. Software Framework for Efficient Tensor Representation and Decompositions for Pattern Recognition in Computer Vision. In Advances in Soft Computing Series "Image Processing & Communications Challenges II", Springer-Verlag, pp. 185–192; 2010.

[6] Duda RO, Hart PE, Stork DG. Pattern Classification. Wiley; 2001.

[7] http://i21www.ira.uka.de/image_sequences/

[8] Lathauwer de L. Signal Processing Based on Multilinear Algebra. PhD dissertation, Katholieke Universiteit Leuven; 1997.

[9] Lathauwer de L, Moor de, B, Vandewalle J. On the Best Rank-1 and Rank-(R_1, R_2, …, R_N) Approximation of Higher-Order Tensors, Vol. 21, No. 4, pp. 1324-1342; 2000.

[10] Muti D, Bourennane S. Survey on tensor signal algebraic filtering. Signal Processing 87, pp. 237-249; 2007.

[11] Savas B, Eldén L. Handwritten digit classification using higher order singular value decomposition. Pattern Recognition, Vol. 40, pp. 993-1003; 2007.

[12] Wang H, Ahuja N: Compact Representation of Multidimensional Data Using Tensro Rank-One Decomposition. Proceedings of the 17th International Conference on Pattern Recognition, Vol. 1, pp. 44-47; 2004.

[13] Wang H, Ahuja N. A Tensor Approximation Approach to Dimensionality Reduction. International Journal of Computer Vision, Vol. 76, pp. 217-229; 2008.