International Journal of Physics and Mathematical Sciences ISSN: 2277-2111 (Online)
An Open Access, Online International Journal Available at http://www.cibtech.org/jpms.htm
2014 Vol. 4 (2) April-June, pp. 29-38/Petwal and Bhatt
Research Article

CAS STUDY OF TUCKER DECOMPOSITION

***K.C. Petwal and Vineet Bhatt**
Department of Mathematics, HNB Garhwal University (A Central University), S.R.T. Campus Badshahi Thaul, Tehri Garhwal-249199, Uttarakhand, India
**Author for Correspondence*

ABSTRACT

This paper deals with Computer Algebraic System (CAS) study of Tucker decomposition with non-negative constrained (NTD). First we reviewed some important tensor and matrix algebraic operation. We provide an overview of NTD, algorithms and some MATLAB codes. Tucker decomposition is a useful technique for decomposition of any higher order tensor, which is applicable in more fields of science and mathematics. There are some algorithms for NTD like: HOSVD, HOOI and ALS. ALS algorithm for Tucker decomposition is very useful especially for noiseless data; it can also be used for the initialization of other algorithms for NTD. The HOSVD and HOOI algorithms can also be used for initialization of NTD, especially when loading matrices $A^{(n)}$ are sparse and orthogonal or close to orthogonal.

MSC 2010:15A69; 15A72; 65Y04, 15A03.
Keywords: *Multi-Linear Algebra, Unfolding, Tucker Decomposition, ALS*

INTRODUCTION

Today higher-order tensor is useful in various fields of application. Kolda presented a review paper on all applications of higher-order tensor (Bader and Kolda, 2009). Tensor decomposition is useful for analysis any data. Various techniques of decompose a tensor has been evolved since their invention like HOSVD (Bhatt, 2013), CANDEC/PARAFAC (Bhatt and Kumar, 2014), Tucker decomposition, tensor unfolding etc. All the above techniques of tensor factorization have been developed to analyse some particular issues and hence checking out their impact on a common problem may yield different results. Tucker decomposition is a technique of decompose a tensor which is commonly useful in various fields of application. Tucker decomposition is used in chemical analysis (Henrion, 1994), signal processing (De Lathauwer and Vandewalle, 2004), Muti and Bourennane (2005) have applied to extend Wiener filters in signal processing and examples from psychometrics are provide by Kiers and Mechelen (2001) in their overview of three-way component analysis techniques and more (Bader and Kolda, 2009). In 1963, Tucker proposed a decomposition method for three-way arrays as a multidimensional extension of factor analysis. Tucker also introduces this decomposition in 1964 and 1966. In the intervening years, several authors developed the decomposition for N-way arrays. Tucker decomposition decomposes a tensor into a set of matrices and one small core tensor. Tucker decomposition is evoked by various names, three-mode factor analysis (3NFA/Tucker3) evoked by Tucker in 1966, three-mode PCA (3MPCA) by Kroonenberg and Leeuw in 1980, N-mode PCA by Kapteyn *et al.*, in 1986, HOSVD by Lathauwer in 2000 and in 2002 Vasilescu and Terzopoulos evoked N-mode SVD. In 1966, Tucker introduced three methods for computing Tucker decomposition, but he was somewhat hampered by the computing ability of the day, stating that calculating the eigen-decomposition for a 300×300 matrix may exceed computer capacity. The basic idea is to find those components that best capture the variation in mode-n, independent of the other modes. Tucker presented it only for the three-way case, but the generalization to N way is straightforward.

Some Standard Operations: We are defining some standard operations, which are useful for our future development;

1) Kronecker Product of matrix $A \in \mathbb{R}^{I \times J}$ and $B \in \mathbb{R}^{K \times L}$ is denoted by $(A \otimes B)$ and the resulting matrix order is $(IK) \times (JL)$ (Bhatt and Kumar, 2014).

(a) $(A \otimes B)(C \otimes D) = AC \otimes BD$, and **(b)** $(A \otimes B)^{\dagger} = A^{\dagger} \otimes B^{\dagger}$.

2) Khatri-Rao product is the column wise Kronecker product (Bhatt and Kumar, 2014). The Khatri-Rao product of matrix $A \in \mathbb{R}^{I\times K}$ and $B \in \mathbb{R}^{J\times K}$ is denoted by $A \odot B$ and its resultant matrix order is $(IJ) \times K$.

Proposition 2.1 (Khatri- Rao product): Let $A \in \mathbb{R}^{I\times L}$, $B \in \mathbb{R}^{J\times L}$ and $C \in \mathbb{R}^{K\times L}$ are three matrices. Then Khatri-Rao products are follows:

a. $A \odot B \odot C = (A \odot B) \odot C = A \odot (B \odot C)$,

b. $(A \odot B)^T(A \odot B) = A^TA * B^TB$ and

c. $(A \odot B)^{\dagger} = \left((A^TA) * (B^TB)\right)^{\dagger}(A \odot B)^T$.

***Outer Product of Two Vector*:** The outer product of two vectors yields a matrix and is typically written as $X = ab^T$. For N dimensions outer product, Let $\mathcal{N} = \{1,2 \dots . N\}$ and $a^n \in \mathbb{R}^{I_n}$ for all $n \in N$. Then the outer product of these N vectors is an Nth-order tensor and defined element wise as $(a^1oa^2o \dots \dots oa^N)_{i_1i_2\dots i_N} = a^1i_1a^2i_2 \dots . . a^Ni_N$ for $1 \leq i_n \leq I_n, n \in N$ (Bhatt and Kumar, 2014).

***The n-Mode Product*:** The n-mode product of a tensor $\mathcal{Y} \in \mathbb{R}^{J_1\times J_2\times \dots \times J_N}$ with a matrix $A \in \mathbb{R}^{I\times J_n}$ is denoted by $\mathcal{Y} \times_n A$. The result is of size $J_1 \times J_2 \times \dots . \times J_{n-1} \times I \times J_{n+1} \times \dots . . \times J_N$ and is defined element wise as $(\mathcal{Y} \times_n A)_{j_1\dots j_{n-1}ij_{n+1}\dots j_N} = \sum_{j_n=1}^{J_n} y_{j_1j_2\dots j_N}a_{ij_n}$. There are many ways of considering n-mode multiplication. For example, let $\mathcal{Y} \in \mathbb{R}^{I\times J\times K}$, $B \in \mathbb{R}^{L\times J}$ and $\mathcal{X} = \mathcal{Y} \times_2 B$. One interpretation is that each mode-2 fiber of $\mathcal{X}$ is the result of multiplying the corresponding mode-2 fiber of $\mathcal{Y}$ by B: $x_{i:k} = By_{i:k}$ for each $i = 1,2, \dots, I, and\ K = 1,2, \dots, K$.

Proposition 2.2 (n-mode matrix product): Let $\mathcal{Y} \in \mathbb{R}^{J_1\times J_2\times \dots \times J_N}$ be an N way tensor.

a. Given matrices $A \in \mathbb{R}^{I_m\times J_m}$, $B \in \mathbb{R}^{I_n\times J_n}$,

$\mathcal{Y} \times_m A \times_n B = (\mathcal{Y} \times_m A) \times_n B = (\mathcal{Y} \times_n B) \times_m A \ \ (m \neq n)$.

b. Given matrix $A \in \mathbb{R}^{I\times J_n}$, $B \in \mathbb{R}^{K\times I}$, $\mathcal{Y} \times_n A \times_n B = \mathcal{Y} \times_n (BA)$.

c. Moreover, if $A \in \mathbb{R}^{I\times J_n}$ with full column rank, then $\mathcal{X} = \mathcal{Y} \times_n A \Longrightarrow \mathcal{Y} = \mathcal{X} \times_n A^{\dagger}$.

d. Consequently, if $A \in \mathbb{R}^{I\times J_n}$ is orthonormal, then $\mathcal{X} = \mathcal{Y} \times_n A \Longrightarrow \mathcal{Y} = \mathcal{X} \times_n A^T$.

***Matricization of a Tensor*:** Especially in computations, it is important to transform a tensor into matrix. The matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times \dots \times I_N}$ is defined as follows. Let the ordered sets $\mathcal{R} = \{r_1, r_2 \dots . r_L\}$ and $\mathcal{C} = \{c_1, c_2 \dots . c_M\}$ be a partitioning of the modes $\mathcal{N} = \{1,2, \dots, N\}$. Recall that I_N denotes the sizes of the tensor: $\{I_1, I_2, \dots I_N\}$. The matricized tensor can then be specified by $X_{\mathcal{R}\times\mathcal{C}\times I_N} \in \mathbb{R}^{J\times K}$ with $J = \prod_{n\in\mathcal{R}} I_n$ and $K = \prod_{n\in\mathcal{C}} I_n$. The indices in $\mathcal{R}$ are mapped to the rows and the indices in $\mathcal{C}$ mapped to the columns. Specifically $X_{(\mathcal{R}\times\mathcal{C}:I_N)_{jk}} = x_{i_1,i_2,\dots,i_N}$ with

$$j = 1 + \sum_{l=1}^{L}[(i_{r_l} - 1)\prod_{l'=1}^{l-1} I_{r_{l'}}] \ and\ k = 1 + \sum_{m=1}^{M}[(i_{r_m} - 1)\prod_{m'=1}^{m-1} I_{r_{m'}}].$$

We can matricize a tensor by MATLAB code. Let X is a multidimensional array, and let the sets R and C be defined. Then MATLAB code in Appendix-I converts to a matrix and back again to a tensor. An important special case is whenever $\mathcal{R}$ is a singleton. This means that the fibers of mode n are aligned as the columns of the resulting matrix. The n-mode matricization of tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times \dots \times I_N}$ is a special case of matricization given by

$$X_n \equiv X_{\mathcal{R}\times\mathcal{C}:I_N} with\ \mathcal{R} = \{n\}\ and\ \mathcal{C} = \{1,2, \dots n-1, n+1 \dots N\}.$$

Proposition 2.3 Let $\mathcal{Y} \in \mathbb{R}^{J_1\times J_2\times \dots \times J_N}$ and $\mathcal{N} = \{1,2, \dots, N\}$.

a. If $A \in \mathbb{R}^{I\times J_n}$ then $\mathcal{X} = \mathcal{Y} \times_n A \Leftrightarrow X_n = AY_n$.

b. Let $A^n \in \mathbb{R}^{I_n\times J_n}$ for all $n \in \mathcal{N}$. If $\mathcal{R} = \{r_1, r_2, \dots r_L\}$ and $\mathcal{C} = \{c_1, c_2, \dots, c_M\}$ partition $\mathcal{N}$, then $\mathcal{X} = \mathcal{Y} \times_1 A^1 \times_2 A^2 \times_3 \dots . \times_N A^N \Leftrightarrow X_{(\mathcal{R}\times\mathcal{C}:J_N)} = (A^{r_L} \otimes \dots \otimes A^{r_1})Y_{(\mathcal{R}\times\mathcal{C}:I_N)}(A^{c_M} \otimes \dots \otimes A^{c_1})^T$.

c. Consequently, If $A^n \in \mathbb{R}^{I_n\times J_n}$ for all $n \in \mathcal{N}$, for any specific $n \in \mathcal{N}$, we have

$$\mathcal{X} = \mathcal{Y} \times_1 A^{(1)} \times_2 \dots . \times_N A^{(N)} \Leftrightarrow X_n = A^nY_n\left(A^{(N)} \otimes \dots . \otimes A^{(n+1)} \otimes A^{(n-1)} \otimes \dots . A^{(1)}\right)^T.$$

d. Moreover, if $\mathcal{C} = \{c_1, c_2, \dots, c_M\} \subseteq \mathcal{N}$ and $A^n \in \mathbb{R}^{I_n \times J_n}$ for $n \in \mathcal{C}$, defining $\mathcal{R} = \mathcal{N}/\mathcal{C}$ yields $\mathcal{X} = \mathcal{Y} \times_{c_1} A^{c_1} \times_{c_2} A^{c_2} \times_{c_3} \dots \times_{c_M} A^{c_M} \Leftrightarrow X_{(\mathcal{R}\times\mathcal{C}:K_{\mathcal{N}})} = Y_{(\mathcal{R}\times\mathcal{C}:I_{\mathcal{N}})}(A^{c_M} \otimes \dots \otimes A^{c_1})^T$ with $K_n = \begin{cases} I_n & if\ n \in \mathcal{C} \\ J_n & if\ n \in \mathcal{R} \end{cases}$.

Norm and Inner Product of a Tensor: Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ be a tensor, then its norm is calculated by

$$\|\mathcal{X}\|^2 = \langle \mathcal{X}, \mathcal{X} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \dots \sum_{i_N=1}^{I_N} x^2_{i_1,i_2\dots i_N}\ , \tag{1}$$

and the inner product of any two tensor $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ is calculated by

$$\langle \mathcal{X}, \mathcal{Y} \rangle = vec(\mathcal{X})^T vec(\mathcal{Y}) = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \dots \sum_{i_N=1}^{I_N} x_{i_1,i_2\dots i_N} y_{i_1,i_2\dots i_N}\ . \tag{2}$$

The norm of a tensor can be transformed to a matrix or vector norm by using the matricized or vectorized version of the tensor (Proposition 2.3). The inner product of two rank-1 tensors can be simplified to be the product of the individual dot products of the components (Bhatt and Kumar, 2014),

Proposition 2.4 Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ and $\mathcal{N} = \{1, 2, \dots, N\}$,

a. Let sets $\mathcal{R}$ and $\mathcal{C}$ be a partitioning of $\mathcal{N}$. Then $\|\mathcal{X}\| = \|X_{\mathcal{R}\times\mathcal{C}:I_n}\|_F$.

b. Let $n \in \mathcal{N}$. Then $\|\mathcal{X}\| = \|X_{(n)}\|_F$.

c. $\|\mathcal{X}\| = \|vec(\mathcal{X})\|_2$.

Proposition 2.5 Let $, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$. Then $\|\mathcal{X} - \mathcal{Y}\|^2 = \|\mathcal{X}\|^2 - \|\mathcal{Y}\|^2 - 2\langle \mathcal{X}, \mathcal{Y} \rangle$.

Proposition 2.6

Let $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ with $\mathcal{X} = a^{(1)} o a^{(2)} o \dots o a^{(N)}$ any $\mathcal{Y} = b^{(1)} o b^{(2)} o \dots o b^{(N)}$, then $\langle \mathcal{X}, \mathcal{Y} \rangle = \prod_{n=1}^{N} \langle a^n, b^n \rangle$.

Proposition 2.7 Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ and let Q be a $J \times I_n$ orthonormal matrix. Then $\|\mathcal{X}\| = \|\mathcal{X} \times_n Q\|$.

Proposition 2.8 Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_{n-1} \times J \times I_{n+1} \times \dots \times I_N}$, $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_{n-1} \times K \times I_{n+1} \times \dots \times I_N}$ and $A \in \mathbb{R}^{J \times K}$. Then $\langle \mathcal{X}, \mathcal{Y} \times_n A \rangle = \langle \mathcal{X} \times_n A^T, \mathcal{Y} \rangle$.

***Nonnegative Tucker Decomposition*:** Tucker Decomposition is also known as Tucker3 decomposition or best rank (J, R, P) approximation. Let a third-order tensor $\mathcal{A} \in \mathbb{R}^{I \times T \times Q}$ and three non-negative indices $(J, R, P) \ll (I, T, Q)$, then by Tucker decomposition, we find three component matrices called factor or loading matrices or factors: $A = [\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_J] \in \mathbb{R}^{I \times J}$, $B = [\boldsymbol{b}_1, \boldsymbol{b}_2, \dots, \boldsymbol{b}_R] \in \mathbb{R}^{T \times R}$ and $C = [\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_p] \in \mathbb{R}^{Q \times P}$ and a core tensor $\mathcal{C} \in [c_{jrp}] \in \mathbb{R}^{J \times R \times P}$:

$$\mathcal{A} = \sum_{j=1}^{J} \sum_{r=1}^{R} \sum_{p=1}^{P} c_{jrp}\ (\boldsymbol{a_j} o \boldsymbol{b_r} o \boldsymbol{c_p}) + \mathcal{E}, \tag{3}$$

where $\mathcal{E}$ is an error tensor.

We can represent equation (3) in different types mathematical formulation with respect to different operator, equation (3) is outer product operator of Tucker decomposition. When one use scalar, mode-n multiplication, vector and Kronecker product operator, we find different types of representation of equation (3) respectively:

$$a_{itq} = \sum_{j=1}^{J} \sum_{r=1}^{R} \sum_{p=1}^{P} c_{jrp}\ (a_{ij}\, o\, b_{tr}\, o\, c_{qp}) + e_{itq}\,, \tag{4}$$

it is also called element wise form.

$$\mathcal{A} = \mathcal{C} \times_1 A \times_2 B \times_3 C + \mathcal{E}, \tag{5}$$

$$vec(\mathcal{A}) = vec(A_{(1)}) \cong (C \otimes B \otimes A) vec(\mathcal{C}), \tag{6}$$

where $A_{(1)}$ matricization of tensor $\mathcal{A}$ (Ist mode) and Kronecker product form is

$$\begin{aligned} A_{(1)} &\cong AC_{(1)}(C \otimes B)^T, \\ A_{(2)} &\cong AC_{(2)}(C \otimes B)^T, \\ A_{(3)} &\cong AC_{(3)}(C \otimes B)^T, \end{aligned} \tag{7}$$

where $a_j \in \mathbb{R}^I$, $b_j \in \mathbb{R}^T$, and $c_j \in \mathbb{R}^Q$ are the vector of component matrices A, B and C respectively and c_{jrp} are scaling factors which are the entries of a core tensor $\mathcal{C} \in [c_{jrp}] \in \mathbb{R}^{J \times R \times P}$. If we impose non-negativity constraints the problem of estimating the component matrices and a core tensor is converted into a generalized NMF problem called the non-negative Tucker decomposition. Let we have three matrices and a core tensor, then we can find a Tucker tensor by following MATLAB code (See Appendix-II):

$M = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 6 & 5 & 4 & 3 & 2 \\ 1 & 4 & 5 & 6 & 7 & 5 \end{bmatrix}$, M is a matrix and converting M into G is a core tensor of size $3 \times 2 \times 3$

$G(:,:,1) = \begin{bmatrix} 1 & 2 \\ 7 & 6 \\ 1 & 4 \end{bmatrix}$, $G(:,:,2) = \begin{bmatrix} 3 & 4 \\ 5 & 4 \\ 5 & 6 \end{bmatrix}$, $G(:,:,3) = \begin{bmatrix} 5 & 6 \\ 3 & 2 \\ 7 & 5 \end{bmatrix}$,

$A = \begin{bmatrix} 1 & 2 & 5 \\ 4 & 5 & 6 \\ 1 & 5 & 9 \\ 4 & 5 & 6 \end{bmatrix}$, $B = \begin{bmatrix} 1 & 2 \\ 4 & 5 \\ 1 & 5 \\ 5 & 6 \end{bmatrix}$ $and\ C = \begin{bmatrix} 1 & 5 & 1 \\ 4 & 5 & 2 \\ 5 & 9 & 8 \\ 6 & 9 & 8 \end{bmatrix}$ assume three matrix A, B and C.

T is a Tucker tensor of size 4 x 4 x 4,

$$\mathrm{T}(:,:,1) = \begin{matrix} 814 & 2419 & 1651 & 2954 \\ 1429 & 4258 & 2887 & 5201 \\ 1531 & 4567 & 3088 & 5579 \\ 1429 & 4258 & 2887 & 5201 \end{matrix}, \quad \mathrm{T}(:,:,2) = \begin{matrix} 1194 & 3528 & 2442 & 4306 \\ 2141 & 6356 & 4349 & 7761 \\ 2279 & 6764 & 4631 & 8259 \\ 2141 & 6356 & 4349 & 7761 \end{matrix},$$

$$\mathrm{T}(:,:,3) = \begin{matrix} 2466 & 7380 & 4950 & 9018 \\ 4384 & 13126 & 8794 & 16040 \\ 4606 & 13834 & 9196 & 16910 \\ 4384 & 13126 & 8794 & 16040 \end{matrix}, \quad \mathrm{T}(:,:,4) = \begin{matrix} 2554 & 7630 & 5140 & 9322 \\ 4553 & 13616 & 9149 & 16637 \\ 4787 & 14354 & 9581 & 17543 \\ 4553 & 13616 & 9149 & 16637 \end{matrix}$$

***The Higher-Order Tucker Decomposition* [1]:** The higher-order Tucker decomposition is described as: let a given tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ into an unknown core tensor $\mathcal{C} \in \mathbb{R}^{J_1 \times J_2 \times \dots \times J_N}$ multiplied by a set of N unknown component matrices, $A^{(n)} = \left[\boldsymbol{a}_1^{(n)}, \boldsymbol{a}_2^{(n)}, \dots, \boldsymbol{a}_{J_n}^{(n)}\right] \in \mathbb{R}^{I_n \times J_n}, (n = 1,2,\dots,N)$ is formulated as:

$$\mathcal{A} = \sum_{J_1=1}^{J_1} \sum_{J_2=1}^{J_2} \dots \dots \sum_{J_N=1}^{J_N} c_{j_1,j_2,\dots j_N} \left(\boldsymbol{a}_{J_1}^{(1)} o \boldsymbol{a}_{J_2}^{(2)} o \dots . o \boldsymbol{a}_{J_N}^{(N)}\right) + \mathcal{E} =$$
$$\mathcal{C} \times_1 A^{(1)} \times_2 A^{(2)} \times_3 \dots . \times_N A^{(N)} + \mathcal{E} = \mathcal{C} \times \{A\} + \mathcal{E} = \hat{\mathcal{A}} + \mathcal{E}, \qquad (8)$$

where tensor $\hat{\mathcal{A}}$ is an approximation tensor of $\boldsymbol{\mathcal{A}}$, and tensor $\mathcal{E} = \mathcal{A} - \hat{\mathcal{A}}$ denotes the error tensor.

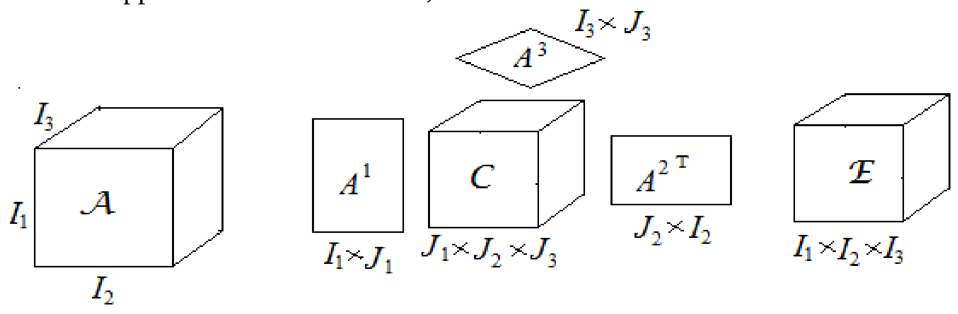

Figure 1: Visualization of Higher Order Tucker Decomposition

***Algorithms for Non-negative Tucker Decomposition*:** Here we discuss some important algorithms of NTD:

HOSVD and HOOI Algorithms: HOSVD and its low-rank counterpart: HOOI (Lathauwer, 2008), (Lathauwer *et al.*, 2000) is useful as an initialization tool for non-negative Tucker decomposition. The HOSVD can be considered as a special form of Tucker decomposition, which decomposes an N-th order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ as

$$\mathcal{A} = \mathcal{C} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 \dots . \times_N U^{(N)}, \qquad (9)$$

where $U^n = \left[\boldsymbol{u}_1^{(n)}, \boldsymbol{u}_2^{(n)}, \dots, \boldsymbol{u}_{J_N}^{(n)}\right] \in \mathbb{R}^{(I_1 \times I_n)}$, $(n = 1,2,\dots,N)$, are orthogonal matrices and the core tensor $\mathcal{C}$ is all orthogonal and ordered tensor of the same dimension as the data tensor $\mathcal{A}$. All-orthogonality means that all sub-tensors $\mathcal{C}_{i_n=k}$ and $\mathcal{C}_{i_n=l}$ obtained by fixing the n-th index to k, l are mutually orthogonal with respect to inner products for all possible value of n, k and l, subject to $k \neq l$, whereas ordering means that

$$\left\|\mathcal{C}_{i_n=1}\right\|_F \geq \left\|\mathcal{C}_{i_n=2}\right\|_F \geq \dots\dots \geq \left\|\mathcal{C}_{i_n=I_n}\right\|_F \forall\, n. \tag{10}$$

This decomposition is a generalization of the standard SVD for the matrix $A \in \mathbb{R}^{I \times T}$,

$$A = UCV^T = C \times_1 U \times_2 V = \sum_{i=1}^{I} \sigma_i\, \boldsymbol{u}_i \boldsymbol{v}_i^T. \tag{11}$$

HOSVD can be written in matrix form as

$$A_{(n)} = U^n C_{(n)} \left(U^{(N)} \otimes \dots . \otimes U^{(n+1)} \otimes U^{(n-1)} \otimes \dots . \otimes U^{(1)}\right)^T, \tag{12}$$

which admits representation in a compact form as

$$A_{(n)} = U^n C_{(n)} V^{(n)T}, \tag{13}$$

where $$V^{(n)} = \left(U^{(N)} \otimes \dots . \otimes U^{(n+1)} \otimes U^{(n-1)} \otimes \dots . \otimes U^{(1)}\right). \tag{14}$$

It then follows that HOSVD can be computed directly in two steps:

1. For $n = 1,2,\dots,N$ compute the unfolded matrices A_n from $\mathcal{A}$ and their standard SVD: $A_{(n)} = U^n C_{(n)} V^{(n)T}$. The orthogonal matrices $U^{(n)}$ are leading left singular vectors of $A_{(n)}$. Alternatively, compute EVD of the covariance matrices; $A_{(n)}^T A_{(n)} = U^n \Lambda^n U^{(n)T}$.
2. Compute the core tensor using the inversion formula

$$\mathcal{C} = \mathcal{A} \times_1 U^{(1)T} \times_2 U^{(2)T} \times_3 \dots . \times_N U^{(N)T}. \tag{15}$$

The HOSVD is computed by means of N standard matrix SVD's and we can reduce the computational cost of HOSVD by using fast and efficient SVD algorithms (Chen and Saad, 2009; Elden and Savas, 2009).

HOSVD results in an ordered orthogonal basis for multi-dimensional representation of input data spanned by each mode of the tensor. In order to achieve dimensionality reduction in each space, we project the data sample onto the principal axis and keep only the component that corresponds to the leading singular values in that subspace. This leads to the concept of the best $R_1, R_2, \dots . R_N$ approximation (Chen and Saad, 2009; Elden and Savas, 2009) formulated as follows: Given a N-th order tensor $\boldsymbol{\mathcal{A}}$, find a lower rank tensor $\hat{\mathcal{A}}$ of same dimension which minimize the FIT

$$\hat{\mathcal{A}} = \arg\ \min_{\hat{\mathcal{A}}} \left\|\mathcal{A} - \hat{\mathcal{A}}\right\|_F^2 . \tag{16}$$

The approximated tensor is represented as

$$\hat{\mathcal{A}} = \tilde{\mathcal{C}} \times_1 \widetilde{U}^{(1)} \times_2 \widetilde{U}^{(2)} \times_3 \dots . \times_N \widetilde{U}^{(N)}, \tag{17}$$

where $\tilde{\mathcal{C}} \in \mathbb{R}^{R_1 \times R_2 \times \dots\dots \times R_N}$ is the core tensor and $\widetilde{U}^{(n)} = \left[\boldsymbol{u}_1^{(n)}, \boldsymbol{u}_2^{(n)}, \dots\dots, \boldsymbol{u}_{R_n}^{(n)}\right] \in \mathbb{R}^{I_n \times R_n}$ are reduced orthogonal matrices with $R_n \leq I_n$ for $n = 1,2,\dots,N$.

A simple approach for solving this problem is to apply a truncated HOSVD, whereby the left singular vectors corresponding to the smallest singular values are ignored. In other words, an approximation with prescribed accuracy can be obtained by appropriate truncation of singular vector of U^n component matrices, corresponding to the singular values below a chosen threshold. Unfortunately, the HOSVD does not attempt to minimize the FIT$\left\|\mathcal{A} - \hat{\mathcal{A}}\right\|_F$, and does not produce an optimal lower rank $R_1, R_2, \dots, R_N$ approximation to $\mathcal{A}$, because it optimizes for each mode separately without taking into account interactions among the modes. However, the HOSVD often produces a close to optimal low rank approximation and is relatively fast in comparison with the iterative algorithms discussed blow (Turney, 2007). The computation of the best rank approximation of a tensor requires an iterative ALS algorithm called HOOI (Lathauwer *et al.*, 2000; Lathauwer and Vandewalle, 2004)

Algorithm-I: HOOI Initialization for NTD

```
Input: $\boldsymbol{\mathcal{A}}$: Input data of size $I_1 \times I_2 \times \ldots.. \times I_N$,
$J_1, J_2, \ldots..., J_N$: Number of basis components for each factor.
Output: $N$ factors $A^{(n)} \in \mathbb{R}_+^{I_n \times J_n}$ and a core tensor $\mathcal{C} \in \mathbb{R}_+^{J_1 \times J_2 \times \ldots.. \times J_N}$, such that the cost function in equation (16) is minimized.
begin
    HOSVD or random initialization for all factors $A^{(n)}$
    repeat
        for n= 1to $N$ do
        $\mathcal{W}^{(-n)} = \mathcal{A} \times_{-n} \{A^T\}$
        $[A^{(n)}, \Sigma^n, V^n] = svds(W_{(n)}^{(-n)}, J_n, 'LM')$        /* $W_{(n)}^{(-n)} \approx A^{(n)} \Sigma^{(n)} V^{(n)T}$ (1*)/
    $A^{(n)} = [A^{(n)}]_+$                                   /* need to fix signs in advance */
        end
    until a stopping criterion is met                /* convergence condition*/
    $\mathcal{G} = \mathcal{W}^{(-N)} \times_N A^{(N)T}$
end
```

((1*) In practice, we use the MATLAB function eigs, i.e., $[A^{(n)}, D_n] = eigs\left(W_{(n)}^{(-n)} W_{(n)}^{(-n)T}, J_n' LM'\right)$, to find the largest magnitude eigenvalues and eigenvectors of a sparse matrix).

The HOOI usually uses the HOSVD to initialize the matrices and the whole procedure can be described as follows. Assuming that the basis orthogonal matrices $\widetilde{U}^{(n)}$ are known or estimated, the core tensor can be obtained as (Lathauwer *et al.*, 2000)

$$\tilde{\mathcal{C}} = \hat{\mathcal{A}} \times_1 \widetilde{U}^{(1)T} \times_2 \widetilde{U}^{(2)T} \times_3 \ldots. \times_N \widetilde{U}^{(N)T}. \qquad (18)$$

Therefore, instead of minimizing equation (16), we can equivalently maximize the cost function.

$$J\left(\widetilde{U}^{(1)}, \widetilde{U}^{(2)}, \ldots.., \widetilde{U}^{(N)}\right) = \left\| \hat{\mathcal{A}} \times_1 \widetilde{U}^{(1)T} \times_2 \widetilde{U}^{(2)T} \times_3 \ldots. \times_N \widetilde{U}^{(N)T} \right\|_F^2, \qquad (19)$$

where only the basis matrices $\widetilde{U}^{(n)}$ are unknown. For example, with $\widetilde{U}^{(1)}, \ldots, \widetilde{U}^{(n-1)}, \widetilde{U}^{(n+1)}, \ldots.., \widetilde{U}^{(N)}$ fixed, we can project tensor $\boldsymbol{\mathcal{A}}$ onto the $\{R_1, R_2, \ldots.., R_{n-1}, R_{n+1}, \ldots..., R_N\}$ - dimensional space defined as

$$\mathcal{W}^{-(n)} = \mathcal{A} \times_1 \widetilde{U}^{(1)T} \ldots \times_{n-1} \widetilde{U}^{(n-1)T} \times_{n+1} \widetilde{U}^{(n+1)T} \ldots. \times_N \widetilde{U}^{(N)T}, \qquad (20)$$

and then the orthogonal matrix $\widetilde{U}^{(n)}$ can be estimated as an orthonormal basis for the dominant subspace of the projection by applying the standard matrix SVD for mode-n unfolding matrix $W_{(n)}^{(-n)}$. The ALS algorithm is used to find the optimal solution for equation (16) (Lathauwer *et al.*, 2000; Lathauwer and Vandewalle, 2004). In each step of the iteration, we optimize only one of the basis matrices, while keeping others fixed. The HOOI algorithm was introduced by Lathauwer *et al.*, (2000) and recently extended and implemented by Bader and Kolda (2009) in their MATLAB tensor toolbox (Bader and Kolda, 2008). The Pseudo-code of the algorithm is described in detail in algorithm-I. To summarize, the idea behind this algorithm is to apply SVD and of find R_n leading left singular vectors of the mode-n matricized version of the product tensor $\mathcal{W}^{(-n)} = \mathcal{A} \times_{-n} \{A^T\}$. By imposing the non-negative constraints on all factor $A^{(n)}$ and with only one or two iterations, the ALS procedure becomes a useful initialization tool for NTD.

The HOOI algorithm generally improves the performance of the best rank approximation as compared to the HOSVD, although it does not always guarantee a globally optimal result. Moreover, it is more computationally demanding than HOSVD since HOOI is an iterative algorithm while the HOSVD is not.

ALS Algorithm for NTD: The NTD is another special kind of the Tucker decomposition (Tucker, 1966), with non-negative constraints, which has already found some applications in neuroscience, bioinformatics and chemometrics (Lathauwer *et al.*, 2000; Morup *et al.*, 2008). There are several existing NTD algorithms (Friendlander and Hatz, 2008; Kim and Choi, 2007; Morup *et al.*, 2008), which process tensors using global learning rules. For large-scale problems, the raw data tensor and its temporary variables stored in memory are very large-scale, and often cause memory overflow error during the decomposition process. One approach to avoid this problem is to process and update the tensor and its factors following block-wise or vector procedures instead of the operation on whole matrices or tensors (Phan and Cichocki, 2008). This approach is referred to as a local decomposition or local learning rule, and will be discussed in the next sections.

For a global ALS algorithm with nonlinear projection, the Tucker decomposition for an N-th order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ can be written in a matricized form as

$$A_{(n)} = A^{(n)} C_{(n)} A^{\otimes_{-n} T} = A^{(n)} C_{(n)} Z_{(-n)} \; (n = 1,2, \ldots, N), \qquad (21)$$

or equivalently in the vectorized form as

$$vec(A_{(n)}) = vec(A^{(n)} C_{(n)} A^{\otimes_{-n} T}) = (A^{\otimes_{-n}} \otimes A^{(n)}) vec(C_{(n)}), \qquad (22)$$

where $Z_{(-n)} = A^{\otimes_{-n} T} = \left[A^{(N)} \otimes \ldots . \otimes A^{(n+1)} \otimes A^{(n-1)} \otimes \ldots . \otimes A^{(1)}\right]^T$. (23)

The above representation forms a basis for the derivation of various updating algorithms for NTD (Kim *et al.*, 2008; Phan and Cichocki 2008). For example, by minimizing alternatively the cost function $\|A_{(n)} -$ *A(n)Z(−n)* for *n=1,2,…N*, and finding fixed points, we obtain the ALS algorithm

$$A^{(n)} \leftarrow \left[A^{(n)} (C_{(n)} Z_{(-n)})^{\dagger}\right]_+, (n = 1,2, \ldots, N), \qquad (24)$$

$$C_{(n)} \leftarrow [A^{(n)\dagger} A_{(n)} Z_{(-n)}^{\dagger}]_+, \qquad (25)$$

for which the simplified form is given by

$$A^{(n)} \leftarrow \left[A_{(n)} \{A^{\dagger}\}^{\otimes_{-n}} C_{(n)}^{\dagger}\right]_+, \; n = 1,2, \ldots, N, \qquad (26)$$

$$\mathcal{C} \leftarrow \left[\mathcal{A} \times \{A^{\dagger}\}\right]_+ = \mathcal{A} \times_1 A^{(1)\dagger} \times_2 A^{(2)\dagger} \ldots . \times_N A^{(N)\dagger}. \qquad (27)$$

HOSVD, HOOI and ALS Algorithms as Initialization Tools for Nonnegative Tensor Decomposition:

The ALS algorithm for Tucker decomposition is very useful, as it can also be used for the initialization of other algorithms for NTD. The HOSVD and HOOI algorithms can also be used for initialization of NTD, especially when loading matrices $A^{(n)}$ are sparse and orthogonal or close to orthogonal. The basic idea of the HOSVD and HOOI initialization of NTD is to apply SVD to unfolded matrices $A_{(n)}$ or a mode-n matricized version of the product tensor $\mathcal{W}^{(-1)} = \mathcal{A} \times_{-n} \{A^T\}$, find $R_n = J_n$ leading left singular vectors of the unfolded matrices, and finally impose the non-negativity constraints on all factors $A^{(n)}$(typically only one or two iterations is sufficient). See pseudo code listing in Algorithms 2 and 3. The advantage of the HOSVD and HOOI approaches over standard ALS is that they can estimate the dimension of the core tensor by analyzing the distribution of singular values.

Algorithm-2: HOSVD initialization

Input: $\boldsymbol{\mathcal{A}}$**:** input data of size $I_1 \times I_2 \times \ldots . \times I_N$,
$J_1, J_2, \ldots, J_N$: Number of basis components for each factor.
Output: N factors $A^{(n)} \in \mathbb{R}_+^{I_n \times J_n}$ and a core tensor $\mathcal{C} \in \mathbb{R}_+^{J_1 \times J_2 \times \ldots \times J_N}$**,** such that the cost function in equation (16) is minimized.

```
begin
for n =1 to $N$ do
        $[A^{(n)}, S^{(n)}, V^{(n)}] = svds(A_{(n)}, J_n, 'L')$
        $A^{(n)} = [A^{(n)}]_+$
end
        $\mathcal{C} = \mathcal{A} \times \{A\}^T$
end
```

Algorithm-3: HOOI initialization:

```
Input: $\boldsymbol{\mathcal{A}}$: input data of size $I_1 \times I_2 \times \dots . \times I_N$,
$J_1, J_2, \dots . , J_N$: Number of basis components for each factor.
Output: $N$ factors $A^{(n)} \in \mathbb{R}_+^{I_n \times J_n}$ and a core tensor $\mathcal{C} \in \mathbb{R}_+^{J_1 \times J_2 \times \dots . \times J_N}$, such that the cost function in equation (16) is minimized.
begin
        $[\{A\}, \mathcal{C}] = HOSVD(\mathcal{A}, [J_1, J_2, \dots . , J_N])$
for n = 1 to $N$ do
$\mathcal{W}^{(-n)} = \mathcal{A} \times_{-n} \{A^T\}$,
$[A^{(n)}, D_n] = eigs(W_{(n)}^{(-n)} W_{(n)}^{(-n)}, J_n, 'LM')$
        $A^{(n)} = [A^{(n)}]_+$
end
$\mathcal{C} = \mathcal{W}^{(-N)} \times_N A^{(N)T}$
end
```

Conclusion

N-way Tucker decomposition, decomposition a higher-order tensor into an unknown core tensor multiplied by a set of N unknown component matrices. It is a fascinating emerging field of research, with many applications. HOSVD and its low-rank counterpart: HOOI is useful as an initialization tool for NTD. The HOSVD can be considered as a special form of Tucker decomposition. The HOOI algorithm generally improves the performance of the best rank approximation as compared to the HOSVD, although it does not always guarantee a globally optimal result. Moreover, it is more computationally demanding than HOSVD since HOOI is an iterative algorithm while the HOSVD is not. The ALS algorithm for Tucker decomposition is very useful, as it can also be used for the initialization of other algorithms for NTD. The HOSVD and HOOI algorithms can also be used for initialization of NTD, especially when loading matrices are sparse and orthogonal or close to orthogonal. The basic idea of the HOSVD and HOOI initialization of NTD is to apply SVD to unfolded matrices $A_{(n)}$ or a mode-n matricized version of the product tensor $\mathcal{W}^{(-1)} = \mathcal{A} \times_{-n} \{A^T\}$, find $R_n = J_n$ leading left singular vectors of the unfolded matrices, and finally impose the nonnegativity constraints on all factors $A^{(n)}$. The advantage of the HOSVD and HOOI approaches over standard ALS is that they can estimate the dimension of the core tensor by analyzing the distribution of singular values.

ACKNOWLEDGMENT

The author(s) express their hearty thanks and gratefulness to all those scientists whose masterpieces have been consulted during the preparation of the present research article. Special thanks are due to Professor Kolda (2009) and Tucker (1966), who's fabulous research work have been entirely consulted to expose the present manuscript.

Appendix-I: (MATLAB code for convert a tensor to matrix and back to tensor)

```
X = rand(1,2,3,4);
R = [2 3];
C = [4 1];
I = size(X);
J = prod(I(R));
K = prod(I(C));
Y = reshape(permute(X,[R C]),J,K); % convert X to matrix Y
Z = ipermute(reshape(Y,[I(R) I(C)]),[R C]); % convert back to tensor
```

Appendix-II: (MATLAB code for make a Tucker tensor)

```
M = [1 2 3 4 5 6;7 6 5 4 3 2; 1 4 5 6 7 5]; %M is a matrix of order (3 by 6)
G = tensor(M, [3 2 3]); %<-- Convert M to as a core tensor.
% Let we find a tensor of order (4 by 4 by 4), then according Fig.(1), set
% the matrices likes A, B and C.
A = [1 2 5 ; 4 5 6 ; 1 5 9 ; 4 5 6]; % Because I1 =4 and J1 = 3.
B = [1 2 ; 4 5 ; 1 5; 5 6];% Because I2 =4 and J2 =2.
C = [1 5 1; 4 5 2; 5 9 8; 6 9 8]; % Because I3 = 4 and J3 =3.
T = ttm(G,{A, B, C}); % We can find Tucker Tensor T.
```

Nomenclature:

S.N.	Notation used	Notation's meaning
1.	o	Outer product of two vector/matrix/tensor
2.	$\otimes$	Kronecker product
3.	$\mathcal{A}, \mathcal{C}, \mathcal{Y}, \mathcal{X}, \mathcal{E} \ldots\ldots\ldots;$	Higher order Tensor
4.	$A, B, C, \ldots\ldots M$	Matrices ($A \in \mathbb{R}^{I \times J}$ representing hidden components)
5.	a, b, …;α, β, ...;a_1, a_2..	Scalars (i.e. t is used for transpose of matrix and vector)
6.	***a,b,c…***	Vectors
7.	$I_1, I_2 \ldots\ldots.; I, J, K$	Indices
8.	*, ·	Product
9.	$\times_n$	n-mode product of tensor by matrix
10.	$\mathbb{R}$	Real vector space
11.	$A_{(n)}, C_{(n)} \ldots.$	Matricization of tensor $\mathcal{A}, \mathcal{C}$ ....
12.	ALS	Alternative least Square
13.	$\odot$	Khatri-Rao Product
14.	$[\ ]_+$	Non-negative Data set
15.	$[\ ]^{\dagger}$	Superscript symbol for Moore-Penrose Pseudo-inversion of a matrix
16.	$[\ ]^T$	Transpose of matrix and vector
17.	HOOI	Higher order orthogonal Iteration
18.	HOSVD	Higher order singular value decomposition
19.	SVD	Singular value decomposition
20.	NTD	Non-negative Tucker decomposition
21.	EVD	Eigen value decomposition
22.	NMF	Non-negative matrix factorization
23.	CAS	Computer Algebraic System

REFERENCE

Bader BW and Kolda TG (2008), MATLAB Tensor Toolbox Version 2.2", January.

Bader BW and Kolda TG (2009). Tensor Decompositions and Applications. *SIAM Review* **51**(3) 455-500.

Chen J and Saad Y (2009). On the tensor SVD and the optimal low rank orthogonal approximation of tensors. *SIAM Journal on Matrix Analysis and Applications* **30** 1709-1734.

De Lathauwer L and Vandewalle J (2004). Dimensionality reduction in higher-order signal processing and rank-(R1, R2, . ., RN) reduction in multilinear algebra. *Linear Algebra and its Applications* **391** 31–55.

Elden L and Savas B (2009). A Newton-Grassmann method for computing the best multi-linear rank-r_1, r_2, r_3 approximation of a tensor. *SIAM Journal on Matrix Analysis and Applications* **31** 248-271.

Friendlander MP and Hatz H (2008), Computing nonnegative tensor factorizations. *Computational Optimization and Applications* **23**(4) 631-647.

Henrion R (1994). N-way principal component analysis theory, algorithms and applications. *Chemometrics and Intelligent Laboratory Systems* **25** 1–23.

Kapteyn A, Neudecker H and Wansbeek T (1986). An approach to n-mode components analysis. *Psychometrika* **51** 269–275.

Kiers HAL and Mechelen IV (2001). Three-way component analysis: Principles and illustrative application. *Psychological Methods* **6** 84–110.
Kim YD and Choi S (2007). Nonnegative Tucker Decomposition", *In Proceeding of Conf. Computer Vision and Pattern Recognition (CVPR-2007), Minneapolis, Minnesota.*
Kim YD *et al.*, (2008). Nonnegative Tucker Decomposition with Alpha Divergence. *In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2008, Nevada, USA.*
Kroonenberg PM and Leeuw J De (1980). Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika* **45** 69–97.
Lathauwer L De (2008). Decompositions of higher order tensor in block terms –part 1: Lemmas for partitioned matrices. *SIAM Journal on Matrix Analysis and Applications* **30**(3) 1022-1032.
Lathauwer L De *et al.*, (2000). ON the best rank-1 and rank- $R_1, R_2, \dots, R_N$ approximation of higher-order tensors. *SIAM Journal of Matrix Analysis and Applications* **21**(4) 1324-1342.
Lathauwer L De, Moor B De and Vandewalle J (2000). A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications* **21** 1253–1278.
Lathauwer L De, Moor B De and Vandewalle J (2000). A multilinear singular value decompositions. *SIAM Journal on Matrix Analysis and Applications* **21**(4) 1253-1278.
Morup M, Hansen LK and Arnfred SM (2008). Algorithms for sparse Nonnegative Tucker Decomposition. *Neural Computation* **20** 2112-2131.
Muti D and Bourennane S (2005). Multidimensional filtering based on a tensor approach. *Signal Process* **85** 2338–2353.
Phan AH and Cichocki A (2008) Fast and efficient algorithms for nonnegative Tucker decomposition. *In Proceeding of the Fifth International Symposium on Neural Networks, Springer LNCS-5264, Beijing, China* 772-782, 24-28.
Tucker LR (1963). Implications of factor analysis of three-way matrices for measurement of change. In: *Problems in Measuring Change* edited by CW Harris (University of Wisconsin Press) 122-37.
Tucker LR (1964). The extension of factor analysis to three-dimensional matrices. In: *Contributions to Mathematical Psychology* edited by H Gulliksen and N Frederiksen (Holt, Rinehardt, & Winston, New York) 110–127.
Tucker LR (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika* **31** 279–311.
Turney PD (2007). Empirical evaluation of four tensor decomposition algorithm. Technical report, National Research Council", Institute for Information Technology, Technical Report ERB-1152.
Vasilescu MAO and Terzopoulos D (2002). Multilinear analysis of image ensembles: TensorFaces. In: ECCV 2002: *Proceedings of the 7th European Conference on Computer Vision, Lecture Notes in Computer Science* 2350, Springer, pp. 447–460.
Vineet Bhatt (2013). Link between HOSVD of a Tensor and SVD of Matrix Unfolding. *International Journal of Physical, Chemical & Mathematical Sciences* **2**(1) 128-139. ISSN: 2278-683X,.
Vineet Bhatt and Kumar S (2014). A CAS Aided Survey of CP-Decomposition and Rank-1 Approximation of a 3rd-Order tensor. Accepted for publication in *Palestine Journal of Mathematics* 2.