Algorithms for Nonnegative Tensor Factorization

Markus Flatz

Technical Report 2013-05 November 2013

Department of Computer Sciences

Jakob-Haringer-Straße 2
5020 Salzburg
Austria
www.cosy.sbg.ac.at

Technical Report Series

Algorithms for Nonnegative Tensor Factorization

Markus Flatz
Department of Computer Sciences
University of Salzburg
Salzburg, Austria
mflatz@cosy.sbg.ac.at

ABSTRACT
Nonnegative Matrix Factorization (NMF) is an efficient technique to approximate a large matrix containing only nonnegative elements as a product of two nonnegative matrices of significantly smaller size. The guaranteed nonnegativity of the factors is a distinctive property that other widely used matrix factorization methods do not have.

Matrices can also be seen as second-order tensors. For some problems, it is necessary to process tensors of third or higher order. For this purpose, NMF can be generalized to Nonnegative Tensor Factorization (NTF). NMF and NTF are used in various application areas, for example in document classification and multi-way data analysis.

The aim of this report is to give an overview over some algorithms to compute Nonnegative Tensor Factorizations, including two multiplicative algorithm based on the Alpha-divergence and the Beta-divergence, respectively, two Hierarchical Alternating Least Squares algorithms and a Block Principal Pivoting algorithm utilizing matricization.

KEY WORDS
High Performance Computing, Nonnegative Tensor Factorization, Nonnegative Matrix Factorization

1 Introduction

A characteristic property of modern society is the increasing need to process large amounts of data. One important class of data is represented by nonnegative matrices and tensors, which occur in many application areas. These are often considerably large, which makes their processing and evaluation difficult and time-consuming.

Nonnegative Matrix Factorization, (abbreviated as NMF or NNMF), is a technique to approximate a nonnegative matrix as a product of two nonnegative matrices. The two resulting matrices are usually smaller than the original matrix and therefore easier to handle and process. In the last decade, NMF has become quite popular and has been applied to a wide variety of practical problems.

The idea of such a factorization was published in 1994 under the name "Positive Matrix Factorization" [21]. In 1999, an article in Nature [14] about Nonnegative Matrix Factorization caught the attention of a wide audience. Several papers were written about NMF since then, discussing its properties, algorithms, modifications and often also possible applications. Some of the various areas where Nonnegative Matrix Factorization was successfully applied are text mining [23] [24] [1], classification of documents [2] and emails [10], clustering [28] [18], spectral data analysis [11] [22] [1], face recognition [29], distance estimation in networks [19], the analysis of EEG data [16], separation of sound sources [27], music transcription [25] [26], computational biology, for example molecular pattern discovery and class comparison and prediction [3] [8] [7] and neuroscience [6].

In contrast to other methods such as singular value decomposition (SVD) or principal component analysis (PCA), NMF has the distinguishing property that the factors are guaranteed to be nonnegative, which allows to view the factorization as an additive combination of features.

2 The NMF Problem

An informal description of the NMF problem is: Given a nonnegative matrix $\mathbf{Y}$ of size $I \times T$, find two nonnegative matrices $\mathbf{A}$ (size $I \times J$) and $\mathbf{X}$ (size $J \times T$) such that their product $\mathbf{AX}$ approximates $\mathbf{Y}$. Figure 1 illustrates the NMF problem for $I = 7$, $T = 9$ and $J = 3$.

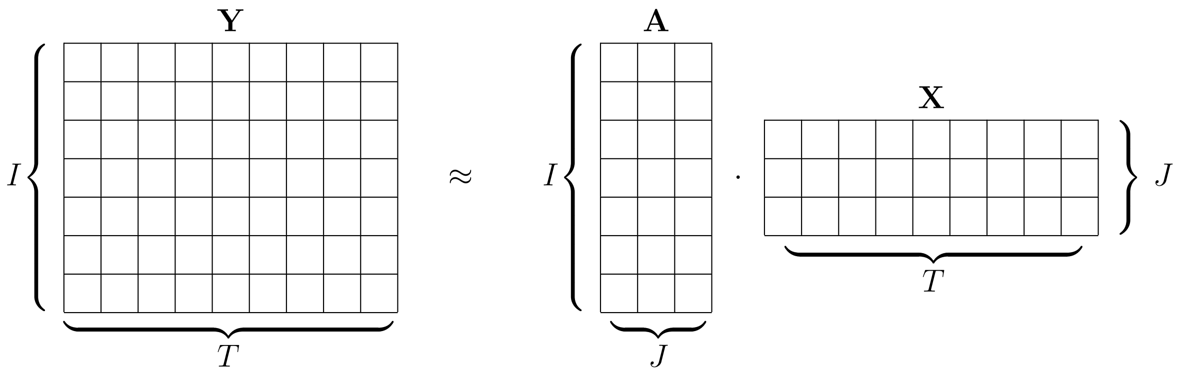

Figure 1: Illustration of the NMF problem

A matrix is called nonnegative if all its elements are ≥ 0. In practical cases, the chosen J is usually much smaller than I and T. It should be noted that, in general, it is not possible to find $\mathbf{A}$ and $\mathbf{X}$ such that $\mathbf{AX} = \mathbf{Y}$. Hence, NMF is "only" an approximation, for this reason it is sometimes called Approximative Nonnegative Matrix Factorization or Nonnegative Matrix Approximation. This is sometimes expressed as $\mathbf{Y} = \mathbf{AX} + \mathbf{E}$, where $\mathbf{E}$ is a matrix of size $I \times T$ that represents the approximation error. Thus, $\mathbf{AX}$ can be seen as a compressed representation of $\mathbf{Y}$, with a rank of j or less.

Formally, NMF can be defined as [20]:

Definition (NMF)**.** *Given a nonnegative matrix* $\mathbf{Y} \in \mathbb{R}^{I\times T}$ *and a positive integer* J*, find nonnegative matrices* $\mathbf{A} \in \mathbb{R}^{I\times J}$ *and* $\mathbf{X} \in \mathbb{R}^{J\times T}$ *that minimize the functional*

$$f(\mathbf{A}, \mathbf{X}) = \frac{1}{2}||\mathbf{Y} - \mathbf{AX}||_F^2 .$$

In [20], $J < \min\{I, T\}$ is explicitly required, this is not strictly necessary, but true in almost all practical cases. For an $I \times T$ matrix $\mathbf{M}$, $||\mathbf{M}||_F$ is the Frobenius norm of $\mathbf{M}$, defined as

$$||\mathbf{M}||_F := \sqrt{\sum_{i=1}^{I}\sum_{t=1}^{T} m_{i,t}^2}$$

where $m_{i,t}$ denotes the element of $\mathbf{M}$ with row index i and column index t. Therefore, $f(\mathbf{A}, \mathbf{X})$ is the square of the Euclidean distance between $\mathbf{Y}$ and $\mathbf{AX}$ with an additional factor $\frac{1}{2}$. The problem is convex in $\mathbf{A}$ and in $\mathbf{X}$ separately, but not in both simultaneously [9].

We note that it is also possible to use other measures to express the distance between $\mathbf{Y}$ and $\mathbf{AX}$, for example the Kullback-Leibler divergence [15], Csiszár's divergences [5] or Alpha- and Beta-divergences [6]. Different measures yield different NMF algorithms, or at least different update steps for the algorithms.

Every column of $\mathbf{A}$ can be interpreted as a basis feature of size I. In total, $\mathbf{A}$ contains J basis features. The multiplication of $\mathbf{A}$ with the nonnegative matrix $\mathbf{X}$ yields a nonnegative matrix $\mathbf{AX}$, where every column of $\mathbf{AX}$ is an additive (or non-subtractive) combination of weighted basis features (columns of $\mathbf{A}$). The famous paper on NMF in Nature [14] uses NMF to represent faces as additive combinations of local parts such as eyes, nose, mouth, etc. However, it was shown in [17] that NMF does not always find such localized features.

3 The NTF Problem

Matrices are second-order tensors. For some applications, for example in multi-way data analysis, the input data are tensors of third or higher order. Therefore, it is desirable to generalize Nonnegative Matrix Factorization to Nonnegative Tensor Factorization.

3.1 Notation

For the formulation of the NTF problem and the algorithms, the following symbols are used:

$\circ$	outer product
$\odot$	Khatri-Rao product
$\circledast$	Hadamard product
$\oslash$	element-wise division
$\times_n$	mode-n product of tensor and matrix
$\mathbf{A}^{\odot_{-n}}$	$= \mathbf{A}^{(N)} \odot \ldots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \ldots \odot \mathbf{A}^{(1)}$

3.2 Problem definition

The Nonnegative Tensor Factorization problem can be formulated as nonnegative canonical decomposition / parallel factor decomposition (CANDECOMP / PARAFAC) as follows (after [6]):

Definition (NTF)**.** *Given an N-th order tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ and a positive integer J, factorize $\underline{\mathbf{Y}}$ into a set of N nonnegative component matrices $\mathbf{A}^{(n)} = [\mathbf{a}_1^{(n)}, \mathbf{a}_2^{(n)}, \ldots, \mathbf{a}_J^{(n)}] \in \mathbb{R}^{I_n \times J}$, ($n = 1, 2, \ldots, N$) representing the common (loading) factors, that is,*

$$\underline{\mathbf{Y}} = \hat{\underline{\mathbf{Y}}} + \underline{\mathbf{E}} = \sum_{j=1}^{J} \mathbf{a}_j^{(1)} \circ \mathbf{a}_j^{(2)} \circ \ldots \circ \mathbf{a}_j^{(N)} + \underline{\mathbf{E}} =$$

$$\underline{\mathbf{I}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)} + \underline{\mathbf{E}} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!] + \underline{\mathbf{E}}$$

with $||\mathbf{a}_j^{(n)}||_2 = 1$ for $n = 1, 2, \ldots N-1$ and $j = 1, 2, \ldots, J$.

The tensor $\underline{\mathbf{E}}$ is the approximation error. Figure 2 illustrates the decomposition for a third-order tensor.

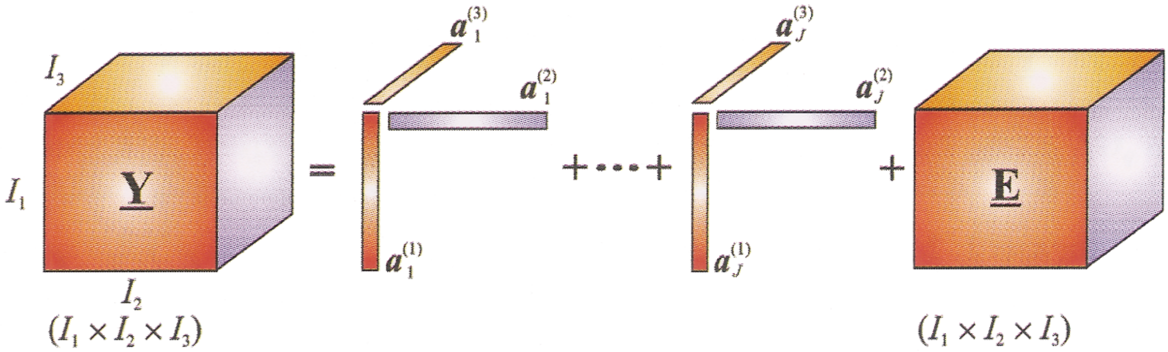

Figure 2: The NTF model for a third-order tensor (from [6])

4 Alpha NTF

The first algorithm presented here uses multiplicative updates based on the Alpha-divergence [6]. Multiplicative algorithms are relatively simple, but can be slower in comparison to enhanced HALS algorithms.

Algorithm 1: Alpha NTF (from [6])

Input: $\underline{\mathbf{Y}}$: input data of size $I_1 \times I_2 \times \ldots \times I_N$, J: number of basis components
Output: N component matrices $\mathbf{A}^{(n)} \in \mathbb{R}_+^{I_n \times J}$

begin
  ALS or random initialization for all factors $\mathbf{A}^{(n)}$;
  $\mathbf{A}_l^{(n)} = \mathbf{A}^{(n)} diag\{\mathbf{1}^T \mathbf{A}^{(n)}\}^{-1}$ for $\forall n$;   `/* normalize to unit length */`
  $\mathbf{A}^{(n)} = \mathbf{A}_l^{(n)}$ for $\forall n \neq N$;
  repeat
    $\hat{\underline{\mathbf{Y}}} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]$;
    for $n = 1$ *to* N **do**
      $$\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)} \circledast \left[\left(\mathbf{Y}_{(n)} \oslash \hat{\mathbf{Y}}_{(n)} \right)^{.[\alpha]} \mathbf{A}_l^{\odot -n} \right]^{.[1/\alpha]};$$
      $\mathbf{A}_l^{(n)} = \mathbf{A}^{(n)} diag\{\mathbf{1}^T \mathbf{A}^{(n)}\}^{-1}$;   `/* normalize to unit length */`
      if $n \neq N$ **then**
        $\mathbf{A}^{(n)} = \mathbf{A}_l^{(n)}$;
      end
    end
  until *a stopping criterion is met*;
end

---

5 Beta NTF

It is also possible to formulate a multiplicative Nonnegative Tensor Factorization algorithm based on the Beta-divergence [6]:

Algorithm 2: Beta NTF (from [6])

Input: $\underline{\mathbf{Y}}$: input data of size $I_1 \times I_2 \times \ldots \times I_N$, J: number of basis components

Output: N component matrices $\mathbf{A}^{(n)} \in \mathbb{R}_+^{I_n \times J}$

begin
    ALS or random initialization for all factors $\mathbf{A}^{(n)}$;
    repeat
        $\hat{\underline{\mathbf{Y}}} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]$;
        for $n = 1$ *to* N **do**
            $\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)} \circledast \left[\left(\mathbf{Y}_{(n)} \oslash \hat{\mathbf{Y}}_{(n)}^{\cdot[\beta-1]}\right) \mathbf{A}^{\odot-n}\right] \oslash \left(\hat{\mathbf{Y}}_{(n)}^{\cdot[\beta]} \mathbf{A}^{\odot-n}\right)$;
            if $n \neq N$ **then**
                $\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)} diag\{\mathbf{1}^T \mathbf{A}^{(n)}\}^{-1}$; `/* normalize */`
            end
        end
    until *a stopping criterion is met*;
end

6 HALS NTF

HALS stands for Hierarchical Alternating Least Squares. The idea is to minimize a set of local cost functions with the same global minima by approximating rank-one tensors [4]. Two algorithms are presented here, first a simple HALS algorithm (Algorithm 3) and then another algorithm that reduces computation of the expensive Khatri-Rao products (Algorithm 4) [6]. Both algorithms use the squared Euclidean distance, but it is also possible to formulate HALS algorithms based on Alpha- and Beta-divergences.

Algorithm 3: Simple HALS NTF (from [6])

Input: $\underline{\mathbf{Y}}$: input data of size $I_1 \times I_2 \times \ldots \times I_N$, J: number of basis components
Output: N component matrices $\mathbf{A}^{(n)} \in \mathbb{R}_+^{I_n \times J}$

begin
  ALS or random initialization for all factors $\mathbf{A}^{(n)}$;
  $\mathbf{a}_j^{(n)} \leftarrow \mathbf{a}_j^{(n)} / ||\mathbf{a}_j^{(n)}||_2$ for $\forall j, n = 1, 2, \ldots, N-1$;   `/* normalize to unit length */`
  $\underline{\mathbf{E}} = \underline{\mathbf{Y}} - \hat{\underline{\mathbf{Y}}} = \underline{\mathbf{Y}} - [\![\{\mathbf{A}\}]\!]$;   `/* residual tensor */`
  repeat
    for $j = 1$ *to* J **do**
      $\underline{\mathbf{Y}}^{(j)} = \underline{\mathbf{E}} + [\![\mathbf{a}_j^{(1)}, \mathbf{a}_j^{(2)}, \ldots, \mathbf{a}_j^{(N)}]\!]$;
      for $n = 1$ *to* N **do**
        $\mathbf{a}_j^{(n)} \leftarrow \left[\mathbf{Y}_{(n)}^{(j)} \{\mathbf{a}_j^{\odot - n}\}\right]_+$;
        if $n \neq N$ **then**
          $\mathbf{a}_j^{(n)} \leftarrow \mathbf{a}_j^{(n)} / ||\mathbf{a}_j^{(n)}||_2$;   `/* normalize to unit length */`
        end
      end
      $\underline{\mathbf{E}} = \underline{\mathbf{Y}}^{(j)} - [\![\mathbf{a}_j^{(1)}, \mathbf{a}_j^{(2)}, \ldots, \mathbf{a}_j^{(N)}]\!]$;
    end
  until *a stopping criterion is met*;
end

Algorithm 4: FAST HALS NTF (from [6])

Input: $\underline{\mathbf{Y}}$: input data of size $I_1 \times I_2 \times \ldots \times I_N$, J: number of basis components
Output: N component matrices $\mathbf{A}^{(n)} \in \mathbb{R}_{+}^{I_n \times J}$

begin
    ALS or random initialization for all factors $\mathbf{A}^{(n)}$;
    $\mathbf{a}_j^{(n)} \leftarrow \mathbf{a}_j^{(n)} / ||\mathbf{a}_j^{(n)}||_2$ for $\forall j, n = 1, 2, \ldots, N-1$; `/* normalize to unit length */`
    $\mathbf{T}^{(1)} = (\mathbf{A}^{(1)T}\mathbf{A}^{(1)}) \circledast \cdots \circledast (\mathbf{A}^{(N)T}\mathbf{A}^{(N)})$;
    repeat
        $\gamma = diag(\mathbf{A}^{(N)T}\mathbf{A}^{(N)})$;
        for $n = 1$ *to* N **do**
            if $n = N$ **then**
                $\gamma = 1$;
            end
            $\mathbf{T}^{(2)} = \mathbf{Y}_{(n)}\{\mathbf{A}^{\odot_{-n}}\}$;
            $\mathbf{T}^{(3)} = \mathbf{T}^{(1)} \oslash (\mathbf{A}^{(n)T}\mathbf{A}^{(n)})$;
            for $j = 1$ *to* J **do**
                $\mathbf{a}_j^{(n)} \leftarrow [\gamma_j \mathbf{a}_j^{(n)} + \mathbf{t}_j^{(2)} - \mathbf{A}^{(n)}\mathbf{t}_j^{(3)}]_+$;
                if $n \neq N$ **then**
                    $\mathbf{a}_j^{(n)} \leftarrow \mathbf{a}_j^{(n)} / ||\mathbf{a}_j^{(n)}||_2$; `/* normalize to unit length */`
                end
            end
            $\mathbf{T}^{(1)} = \mathbf{T}^{(3)} \circledast (\mathbf{A}^{(n)T}\mathbf{A}^{(n)})$;
        end
    until *a stopping criterion is met*;
end

7 Block Principal Pivoting

Another way to compute a Nonnegative Tensor Factorization is called Block Principal Pivoting, and is an active-set-like method [12] [13].

A tensor $\underline{\mathbf{Y}}$ of size $I_1 \times I_2 \times \ldots \times I_N$ can be transformed into a matrix $\mathbf{Y}_{MAT(n)}$ in a process called mode-n-matricization by linearizing all indices except n in the following way: $\mathbf{Y}_{MAT(n)}$ is a matrix of size $I_n \times \prod_{k=1,k\neq n}^{N} I_k$ and the $(i_1, \ldots, i_N)$th element of $\underline{\mathbf{Y}}$ is mapped to the (i_n, L)th element of $\mathbf{Y}_{MAT(n)}$ where

$$L = 1 + \sum_{k=1}^{N} (i_k - 1) L_k$$

and

$$L_k = \prod_{j=1,j\neq n}^{k-1} I_j$$

For a tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, its approximation $\hat{\underline{\mathbf{Y}}} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]$ can be written as, for any $n \in 1, \ldots, N$

$$\mathbf{Y}_{MAT(n)} = \mathbf{A}^{(n)} \times \left(\mathbf{A}^{\odot -n}\right)^T$$

which can be utilized in the ANLS (alternating nonnegativity-constrained least squares) framework. The following subproblems have to be solved:

$$min_{\mathbf{A}^{(n)} \geq \mathbf{0}} \left|\left| \mathbf{A}^{\odot -n} \times \left(\mathbf{A}^{(n)}\right)^T - \left(\mathbf{Y}_{MAT(n)}\right)^T \right|\right|_F^2$$

Algorithm 5 shows the Block Principal Pivoting algorithm that can be used to solve this problem. Since the problem was reduced to a matrix problem, any NMF algorithm could be used to compute the factorization, but algorithms that exploit the special properties of the matrices are especially promising. ($\mathbf{A}^{\odot -n}$ is typically long and thin, while $\left(\mathbf{A}^{(n)}\right)^T$ is typically flat and wide.) In the formulation of the Block Principal Pivoting algorithm below, $\mathbf{x}_{\mathcal{F}_l}$ and $\mathbf{y}_{\mathcal{G}_l}$ represent the subset of the lth column of $\mathbf{X}$ and $\mathbf{Y}$ indexed by $\mathcal{F}_l$ and $\mathcal{G}_l$, respectively.

Algorithm 5: Block principal pivoting (from [13])

Input: $\mathbf{V} \in \mathbb{R}^{P \times Q}, \mathbf{W} \in \mathbb{R}^{P \times L}$
Output: $\mathbf{X}(\in \mathbb{R}^{Q \times L} = arg(min_{\mathbf{X} \geq \mathbf{0}} ||\mathbf{VX} - \mathbf{W}||_F^2))$

begin
  Compute $\mathbf{V}^T\mathbf{V}$ and $\mathbf{V}^T\mathbf{W}$;
  Initialize $\mathcal{F}_l = \emptyset$ and $\mathcal{G}_l = \{1, \ldots, q\}$ for all $l \in 1, \ldots, L$;
  $\mathbf{X} = \mathbf{0}$;
  $\mathbf{Y} = -\mathbf{V}^T\mathbf{W}$;
  $\alpha(\in \mathbb{R}^r) = \mathbf{3}$;
  $\beta(\in \mathbb{R}^r) = (\mathbf{q} + \mathbf{1})$;
  Compute all $\mathbf{x}_{\mathcal{F}_l}$ using column grouping from $\mathbf{V}_{\mathcal{F}_l}^T \mathbf{V}_{\mathcal{F}_l} \mathbf{x}_{\mathcal{F}_l} = \mathbf{V}_{\mathcal{F}_l}^T \mathbf{w}_l$;
  Compute all $\mathbf{y}_{\mathcal{G}_l} = \mathbf{V}_{\mathcal{G}_l}^T (\mathbf{V}_{\mathcal{F}_l} \mathbf{x}_{\mathcal{F}_l} - \mathbf{w}_l)$;
  /* $(\mathbf{x}_{\mathcal{F}_l}, \mathbf{y}_{\mathcal{G}_l})$ is feasible iff $\mathbf{x}_{\mathcal{F}_l} \geq \mathbf{0}$ and $\mathbf{y}_{\mathcal{G}_l} \geq \mathbf{0}$ */
  while *any* $(\mathbf{x}_{\mathcal{F}_l}, \mathbf{y}_{\mathcal{G}_l})$ *is infeasible* **do**
    Find the indices of columns in which the solution is infeasible:
    $I = \{j : (\mathbf{x}_{\mathcal{F}_l}, \mathbf{y}_{\mathcal{G}_l}) \text{ is infeasible}\}$;
    forall the $l \in I$ **do**
      $\mathcal{H}_l = \{q \in \mathcal{F}_l : \mathbf{x}_q < 0\} \cup \{q \in \mathcal{G}_l : \mathbf{y}_q < 0\}$;
    end
    forall the $l \in I$ *with* $|\mathcal{H}_l| < \beta_l$ **do**
      $\beta_l = |\mathcal{H}_l|$;
      $\alpha_l = 3$;
      $\hat{\mathcal{H}}_l = \mathcal{H}_l$;
    end
    forall the $l \in I$ *with* $|\mathcal{H}_l| \geq \beta_l$ *and* $\alpha_l \geq 1$ **do**
      $\alpha_l = \alpha_l - 1$;
      $\hat{\mathcal{H}}_l = \mathcal{H}_l$;
    end
    forall the $l \in I$ *with* $|\mathcal{H}_l| \geq \beta_l$ *and* $\alpha_l = 0$ **do**
      $\hat{\mathcal{H}}_l = \{q : q = max\{q \in \mathcal{H}_l\}\}$;
    end
    forall the $l \in I$ **do**
      $\mathcal{F}_l = (\mathcal{F}_l - \hat{\mathcal{H}}_l) \cup (\hat{\mathcal{H}}_l \cap \mathcal{G}_l)$;
      $\mathcal{G}_l = (\mathcal{G}_l - \hat{\mathcal{H}}_l) \cup (\hat{\mathcal{H}}_l \cap \mathcal{F}_l)$;
      Compute $\mathbf{x}_{\mathcal{F}_l}$ using column grouping from $\mathbf{V}_{\mathcal{F}_l}^T \mathbf{V}_{\mathcal{F}_l} \mathbf{x}_{\mathcal{F}_l} = \mathbf{V}_{\mathcal{F}_l}^T \mathbf{w}_l$;
      $\mathbf{y}_{\mathcal{G}_l} = \mathbf{V}_{\mathcal{G}_l}^T (\mathbf{V}_{\mathcal{F}_l} \mathbf{x}_{\mathcal{F}_l} - \mathbf{w}_l)$;
    end
  end
end

8 Conclusion

In this report, the Nonnegative Tensor Factorization problem was defined and five algorithms to compute this factorization were presented. For future work, it would be especially interesting to parallelize some of these algorithms to allow the processing of large problems which arise in real-world applications on highly parallel modern computing systems.

References

[1] Michael W. Berry, Murray Browne, Amy N. Langville, V. Paul Pauca, and Robert J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics and Data Analysis*, 52(1):155 – 173, 2007.

[2] Michael W. Berry, Nicolas Gillis, and François Glineur. Document classification using nonnegative matrix factorization and underapproximation. In *International Symposium on Circuits and Systems (ISCAS)*, pages 2782–2785. IEEE, 2009.

[3] Jean-Philippe Brunet, Pablo Tamayo, Todd R. Golub, and Jill P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Sciences of the United States of America*, 101(12):4164, 2004.

[4] Andrzej Cichocki, Anh Huy Phan, and Cesar Caiafa. Flexible HALS Algorithms for Sparse Non-negative Matrix/Tensor Factorization. In *Proceedings of 2008 IEEE International Workshop on Machine Learning for Signal Processing*, pages 73–78, 2008.

[5] Andrzej Cichocki, Rafal Zdunek, and Shun-ichi Amari. Csiszár's divergences for non-negative matrix factorization: Family of new algorithms. In *Independent Component Analysis and Blind Signal Separation*, volume 3889 of *Lecture Notes in Computer Science*, pages 32–39. Springer Berlin / Heidelberg, 2006.

[6] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-Ichi Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation.* John Wiley & Sons, Ltd, 2009.

[7] Karthik Devarajan. Nonnegative matrix factorization: an analytical and interpretive tool in computational biology. *PLoS computational biology*, 4(7):e1000029, 2008.

[8] Yuan Gao and George Church. Improving molecular cancer class discovery through sparse non-negative matrix factorization. *Bioinformatics*, 21(21):3970–3975, 2005.

[9] Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

[10] Andreas Janecek and Wilfried Gansterer. E-mail classification based on non-negative matrix factorization. In *Text Mining 2009*, 2009.

[11] Arto Kaarna. Non-negative matrix factorization features from spectral signatures of AVIRIS images. In *International Conference on Geoscience and Remote Sensing Symposium (IGARSS)*, pages 549–552. IEEE, 2006.

[12] Jingu Kim and Haesun Park. Fast Nonnegative Tensor Factorization with an Active-Set-Like Method. In Michael W. Berry, Kyle A. Gallivan, Efstratios Gallopoulos, Ananth Grama, Bernard Philippe, Yousef Saad, and Faisal Saied, editors, *High-Performance Scientific Computing – Algorithms and Applications*, pages 311–326. Springer, 2012.

[13] Jingu Kim and Haesun Park. Fast Nonnegative Tensor Factorization with an Active-Set-Like Method. In Michael W. Berry, Kyle A. Gallivan, Efstratios Gallopoulos, Ananth Grama, Bernard Philippe, Yousef Saad, and Faisal Saied, editors, *High-Performance Scientific Computing: Algorithms and Applications*, pages 311–326. Springer London, 2012.

[14] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[15] Daniel D. Lee and H. Sebastian Seung. Algorithms for Non-negative Matrix Factorization. *Advances in Neural Information Processing Systems (NIPS)*, 13:556–562, 2001.

[16] Hyekyoung Lee, Andrzej Cichocki, and Seungjin Choi. Nonnegative matrix factorization for motor imagery EEG classification. In *Artificial Neural Networks – ICANN 2006*, volume 4132 of *Lecture Notes in Computer Science*, pages 250–259. Springer Berlin / Heidelberg, 2006.

[17] Stan Z. Li, XinWen Hou, HongJiang Zhang, and QianSheng Cheng. Learning spatially localized, parts-based representation. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, volume 1, pages 207–212. IEEE, 2001.

[18] Tao Li and Chris Ding. The relationships among various nonnegative matrix factorization methods for clustering. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 362–371. IEEE, 2006.

[19] Yun Mao, Lawrence K. Saul, and Jonathan M. Smith. IDES: An Internet Distance Estimation Service for Large Networks. *IEEE Journal on Selected Areas in Communications (JSAC)*, 24(12), 2006.

[20] Gabriel Okša, Martin Bečka, and Marián Vajteršic. Nonnegative Matrix Factorization: Algorithms and Parallelization. Technical Report 2010-05, University of Salzburg, Department of Computer Sciences, 2010.

[21] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

[22] V. Paul Pauca, J. Piper, and Robert J. Plemmons. Nonnegative matrix factorization for spectral data analysis. *Linear Algebra and its Applications*, 416(1):29 – 47, 2006.

[23] V. Paul Pauca, Farial Shahnaz, Michael W. Berry, and Robert J. Plemmons. Text mining using nonnegative matrix factorizations. In *Proceedings of the Fourth SIAM International Conference on Data Mining*, pages 452–456, 2004.

[24] Farial Shahnaz, Michael W. Berry, V. Paul Pauca, and Robert J. Plemmons. Document clustering using nonnegative matrix factorization. *Information Processing and Management*, 42(2):373–386, 2006.

[25] Paris Smaragdis and Judith C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180. IEEE, 2003.

[26] Emmanuel Vincent, Nancy Berlin, and Roland Badeau. Harmonic and inharmonic nonnegative matrix factorization for polyphonic pitch transcription. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 109–112. IEEE, 2008.

[27] Tuomas Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *Transactions on Audio, Speech, and Language Processing*, 15(3):1066–1074, 2007.

[28] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval*, pages 267–273. ACM, 2003.

[29] Stefanos Zafeiriou, Anastasios Tefas, Ioan Buciu, and Ioannis Pitas. Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification. *Transactions on Neural Networks*, 17(3):683–695, 2006.