Convex Nonnegative Matrix Factorization with Rank-1 Update for Clustering

Rafał Zdunek(✉)

Department of Electronics, Wroclaw University of Technology,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland
rafal.zdunek@pwr.wroc.pl

Abstract. In convex nonnegative matrix factorization, the feature vectors are modeled by convex combinations of observation vectors. In the paper, we propose to express the factorization model in terms of the sum of rank-1 matrices. Then the sparse factors can be easily estimated by applying the concept of the Hierarchical Alternating Least Squares (HALS) algorithm which is still regarded as one of the most effective algorithms for solving many nonnegative matrix factorization problems. The proposed algorithm has been applied to find partially overlapping clusters in various datasets, including textual documents. The experiments demonstrate the high performance of the proposed approach.

Keywords: Nonnegative matrix factorization · Convex NMF · HALS algorithm · β-divergence · Partitional clustering

1 Introduction

Nonnegative Matrix Factorization (NMF) [1,2] is an unsupervised learning technique that is commonly used in machine learning and data analysis for feature extraction and dimensionality reduction of nonnegative data. The basic model for NMF assumes a decomposition of a nonnegative input matrix into two lower-rank nonnegative matrices. The one represents nonnegative feature or basis vectors, and the other, referred to as an encoding matrix, contains coefficients of nonnegative combinations of the feature vectors.

Convex NMF (CNMF) is a special case of the standard model in which the feature vectors are expressed by linear combinations of observation vectors. Hence, they lie in the space spanned by observation vectors, and may not be constrained to nonnegative values as in the standard NMF model. This model was first proposed by Ding *et al.* [3] for clustering of unsigned data. It is conceptually closely related to the k-means, however the experiments carried out in [3] demonstrated its superiority over the standard k-means with respect to clustering accuracy. Then, CNMF was further developed and improved.

Thurau *et al.* [4] proposed Convex-Hull NMF (CH-NMF) in which the clusters are restricted to be combinations of vertices of the convex hull formed by observation points. Due to distance preserving low-dimensional embeddings, the

L. Rutkowski et al. (Eds.): ICAISC 2015, Part II, LNAI 9120, pp. 59–68, 2015.
DOI: 10.1007/978-3-319-19369-4_6

vertices can be computed efficiently by formulating the CNMF on projected low-dimensional data. CH-NMF is thus scalable, and can be applied for clustering large-scale datasets. The convex model of NMF is also discussed by Esser *et al.* [5] in the context of endmember identification in hyperspectral unmixing.

The factors in CNMF [3] are updated with multiplicative rules, similarly as in the NMF models proposed by Lee and Seung [1]. The multiplicative algorithms are simple to implement and guarantee non-increasing minimization of the objective function. However, their convergence is terrible slow and unnecessarily towards to the minimum that is optimal according to the Karush-Kuhn-Tucker (KKT) optimality conditions. Hence, there is a need for searching more efficient algorithms for CNMF. Krishnamurthy *et al.* [6] extended CNMF by applying the Projected Gradient (PG) algorithm, which considerably improves the convergence properties. Despite this, the convergence is still linear and it might be a problem with satisfying the KKT optimality conditions in each iterative step.

To considerably improve the convergence properties of CNMF, we propose to apply the concept of the Hierarchical Alternating Least Squares (HALS) algorithm which was first used for NMF by Cichocki *et al.* [7]. In this method, the NMF model is expressed by the sum of rank-1 factors that are updated sequentially, subject to nonnegativity constraints. This approach can be also used for minimization of the α- and β-divergence [2]. To significatively reduce its computational complexity, Cichocki and Phan [8] proposed the Fast HALS, which is a reformulated and considerably improved version of the original HALS. Many independent researches [9,10,11,12,13] confirm its high effectiveness for solving various NMF problems and its very fast convergence.

Motivated by the success of the HALS, we apply this concept to CNMF by expressing the factorization model by the sum of rank-1 factors, both for the standard Euclidean distance and the β-divergence. Then applying the similar transformations as in [8], the computational complexity of the proposed HALS-based algorithms for CNMF is considerably reduced.

The paper is organized as follows: Section 2 discusses the CNMF model. The optimization algorithms for estimating the factors in CNMF are presented in Section 3. The experiments carried out for clustering various datasets are described in Section 4. Finally, the conclusions are drawn in Section 5.

2 Convex NMF

The aim of NMF is to find such lower-rank nonnegative matrices $\boldsymbol{A} = [a_{ij}] \in \mathbb{R}_+^{I \times J}$ and $\boldsymbol{X} = [x_{jt}] \in \mathbb{R}_+^{J \times T}$ that $\boldsymbol{Y} = [y_{it}] \cong \boldsymbol{A}\boldsymbol{X} \in \mathbb{R}_+^{I \times T}$, given the data matrix $\boldsymbol{Y}$, the lower rank J, and possibly some prior knowledge on the matrices $\boldsymbol{A}$ or $\boldsymbol{X}$. The set of nonnegative real numbers is denoted by $\mathbb{R}_+$. When NMF is applied for model dimensionality reduction, we usually assume: $J << \frac{IT}{I+T}$ or at least: $J \leq \min\{I, T\}$.

Assuming each column vector of $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T]$ represents a single observation (a datum point in $\mathbb{R}^I$), and J is *a priori* known number of clusters, we can interpret the feature vectors, i.e. the column vectors of $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_J]$, as the centroids (indicating the directions of central points of clusters in $\mathbb{R}^I$) and the entries in $\boldsymbol{X} = [x_{jt}]$ as indicators to the clusters. Normalizing each $\boldsymbol{x}_t$ to the unit l_1 norm, each x_{jt} can be regarded as the probability of assigning the vector $\boldsymbol{y}_t$ to the j-th cluster. If the clusters are disjoint, $\boldsymbol{X}$ should be a binary matrix [14].

In CNMF, each feature vector $\boldsymbol{a}_j$ is assumed to be a convex combination of the data points, i.e. $\forall j : \boldsymbol{a}_j = \sum_{s=1}^{T} w_{sj} \boldsymbol{y}_s$, where $w_{sj} \in \mathbb{R}_+$ are weighting factors, and $\sum_{s=1}^{T} w_{sj} = 1$. Thus, the CNMF model has the form:

$$\boldsymbol{Y} \cong \boldsymbol{Y}\boldsymbol{W}\boldsymbol{X}, \tag{1}$$

where $\boldsymbol{W} \in \mathbb{R}_+^{T \times J}$, $\boldsymbol{1}_T^T \boldsymbol{W} = \boldsymbol{1}_J^T$, and $\boldsymbol{1}_M = [1, \ldots, 1]^T \in \mathbb{R}^M$ is a M-dimensional vector of all ones.

Each centroid is therefore a weighted sum of observation vectors. If only a few vectors $\{\boldsymbol{y}_s\}$ affect the centroid $\boldsymbol{a}_j$, the vector $\boldsymbol{w}_j = [w_{sj}] \in \mathbb{R}_+^T$ is nonnegative and very sparse. If the clusters are only slightly overlapped, the matrix $\boldsymbol{X}$ is also nonnegative and very sparse. Hence, the nonnegativity and sparsity constraints are typically imposed on the factors $\boldsymbol{W}$ and $\boldsymbol{X}$ in CNMF.

If $\boldsymbol{X} = \boldsymbol{W}^T$ in (1), the model is known as the Cluster NMF [3], and it is suitable for clustering the columns in $\boldsymbol{Y}$. For clustering the rows, the nonlinear projective NMF [15] can be used. It is expressed by the model: $\boldsymbol{Y} = \boldsymbol{W}\boldsymbol{W}^T\boldsymbol{Y}$, where $\boldsymbol{W} = [w_{ij}] \in \mathbb{R}_+^{I \times J}$. If $w_{ij} = 1$ and $\forall m \neq i : w_{mj} = 0$, then i-th row of $\boldsymbol{Y}$ belongs to the j-th cluster.

3 Algorithms

The matrices $\boldsymbol{W}$ and $\boldsymbol{X}$ in (1) can be estimated by minimizing various objective functions. Assuming a normally distributed residual error, the objective function is expressed by the squared Euclidean distance:

$$\Psi(\boldsymbol{W}, \boldsymbol{X}) = \frac{1}{2} ||\boldsymbol{Y} - \boldsymbol{Y}\boldsymbol{W}\boldsymbol{X}||_F^2. \tag{2}$$

Let $\boldsymbol{Y}^T\boldsymbol{Y} = [\boldsymbol{Y}^T\boldsymbol{Y}]^+ - [\boldsymbol{Y}^T\boldsymbol{Y}]^-$, where $[b_{ij}]^+ = \max\{0, b_{ij}\}$ and $[b_{ij}]^- = \max\{0, -b_{ij}\}$. Applying the majorization-minimization approach, Ding *et al.* proposed the following multiplicative updating rules:

$$w_{tj}^{(k+1)} = w_{tj}^{(k)} \sqrt{\frac{\left[\left([\boldsymbol{Y}^T\boldsymbol{Y}]^+ + [\boldsymbol{Y}^T\boldsymbol{Y}]^- \boldsymbol{W}^{(k)}\boldsymbol{X}^{(k)}\right)(\boldsymbol{X}^{(k)})^T\right]_{tj}}{\left[\left([\boldsymbol{Y}^T\boldsymbol{Y}]^- + [\boldsymbol{Y}^T\boldsymbol{Y}]^+ \boldsymbol{W}^{(k)}\boldsymbol{X}^{(k)}\right)(\boldsymbol{X}^{(k)})^T\right]_{tj}}}, \tag{3}$$

$$x_{jt}^{(k+1)} = x_{jt}^{(k)} \sqrt{\frac{\left[(\boldsymbol{W}^{(k+1)})^T \left([\boldsymbol{Y}^T\boldsymbol{Y}]^+ + [\boldsymbol{Y}^T\boldsymbol{Y}]^- \boldsymbol{W}^{(k+1)}\boldsymbol{X}^{(k)}\right)\right]_{jt}}{\left[(\boldsymbol{W}^{(k+1)})^T \left([\boldsymbol{Y}^T\boldsymbol{Y}]^- + [\boldsymbol{Y}^T\boldsymbol{Y}]^+ \boldsymbol{W}^{(k+1)}\boldsymbol{X}^{(k)}\right)\right]_{jt}}}, \tag{4}$$

The computational complexity of the update rule (3) can be roughly estimated as $O(IT^2) + O(kJT^2)$, where the first term concerns the computation of the matrix $\boldsymbol{Y}^T\boldsymbol{Y}$, and k is the number of alternating steps. The rule (4) has the similar cost.

3.1 HALS-Based CNMF

Let the model (1) be expressed by the sum of rank-1 matrices:

$$\boldsymbol{Y} = \sum_{t=1}^{T} \boldsymbol{y}_t \underline{\boldsymbol{w}}_t \boldsymbol{X} = \sum_{t=1}^{T} \boldsymbol{y}_t \underline{\boldsymbol{z}}_t, \tag{5}$$

where $\boldsymbol{y}_t \in \mathbb{R}^I$ is the t-th column vector of $\boldsymbol{Y}$, $\underline{\boldsymbol{w}}_t \in \mathbb{R}^{1\times J}$ is the t-th row vector $\boldsymbol{W}$, and $\underline{\boldsymbol{z}}_t = \underline{\boldsymbol{w}}_t\boldsymbol{X} \in \mathbb{R}^{1\times T}$. Note that $\forall t : \boldsymbol{y}_t\underline{\boldsymbol{z}}_t \in \mathbb{R}_+^{I\times T}$, $\text{rank}(\boldsymbol{y}_t\underline{\boldsymbol{z}}_t) = 1$. Considering the model (5), the objective function in (2) can be rewritten as:

$$\Psi(\boldsymbol{W},\boldsymbol{X}) = \frac{1}{2}||\boldsymbol{Y} - \sum_{r\neq t} \boldsymbol{y}_r\underline{\boldsymbol{w}}_r\boldsymbol{X} - \boldsymbol{y}_t\underline{\boldsymbol{w}}_t\boldsymbol{X}||_F^2 = \frac{1}{2}||\boldsymbol{Y}^{(t)} - \boldsymbol{y}_t\underline{\boldsymbol{w}}_t\boldsymbol{X}||_F^2, \tag{6}$$

where $\boldsymbol{Y}^{(t)} = \boldsymbol{Y} - \sum_{r\neq t} \boldsymbol{y}_r\underline{\boldsymbol{w}}_r\boldsymbol{X}$.

The stationary point of $\Psi(\boldsymbol{W},\boldsymbol{X})$ with respect to $\underline{\boldsymbol{w}}_t$ can be obtained from the condition:

$$\nabla_{\underline{\boldsymbol{w}}_t}\Psi(\boldsymbol{W},\boldsymbol{X}) = -\boldsymbol{y}_t^T\boldsymbol{Y}^{(t)}\boldsymbol{X}^T + \xi_t\underline{\boldsymbol{w}}_t\boldsymbol{X}\boldsymbol{X}^T \triangleq \boldsymbol{0}, \tag{7}$$

where $\xi_t = ||\boldsymbol{y}_t||_2^2$. The closed-form updating rule for $\underline{\boldsymbol{w}}_t$ has the form:

$$\underline{\boldsymbol{w}}_t = \xi_t^{-1}\boldsymbol{y}_t^T\boldsymbol{Y}^{(t)}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{X}^T)^{-1}. \tag{8}$$

The computational complexity of the update rule in (8) depends on its implementation. Let the matrix $\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{X}^T)^{-1}$ and the vectors $\{\xi_t\}$ be precomputed with the approximative costs $O(J^3 + J^2T)$ and $O(IT)$, respectively. Then, the total cost of performing k alternating steps with (8) is about $O(J^3 + J^2T + IT) + kO(IT^2 + JT^2)$. If $J << T$, we have $O(kIT^2)$. Thus the computational complexity is I/J-times higher than for the update rule (3). In practice, the implementation of (8) needs the sweeping over the index t, which involves a nested loop in Matlab but the rule (3) can be fully vectorized. Hence, there is a need to redefining the rule (8) in order to implement it more efficiently (especially in Matlab).

The matrix $\boldsymbol{Y}^{(t)}$ can be rewritten as:

$$\boldsymbol{Y}^{(t)} = \boldsymbol{Y} - \boldsymbol{Y}\boldsymbol{W}\boldsymbol{X} + \boldsymbol{y}_t\underline{\boldsymbol{w}}_t\boldsymbol{X}. \tag{9}$$

Inserting (9) to (8) and assuming $\boldsymbol{X}\boldsymbol{X}^T$ is a full rank matrix, we have:

$$\begin{aligned}\underline{\boldsymbol{w}}_t &\leftarrow \xi_t^{-1}\boldsymbol{y}_t^T\left(\boldsymbol{Y} - \boldsymbol{Y}\boldsymbol{W}\boldsymbol{X} + \boldsymbol{y}_t\underline{\boldsymbol{w}}_t\boldsymbol{X}\right)\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{X}^T)^{-1} \\ &= \underline{\boldsymbol{w}}_t + \xi_t^{-1}\boldsymbol{y}_t^T\boldsymbol{Y}\left(\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{X}^T)^{-1} - \boldsymbol{W}\right).\end{aligned} \tag{10}$$

The matrices $\boldsymbol{Y}^T\boldsymbol{Y}$ and $\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{X}^T)^{-1}$ can be precomputed. Thus the overall computational complexity for k iterations of (10) is $O(J^3 + J^2T + IT^2) + O(kJT^2)$, which is at least I/J-times lower than for (8).

The rule (10) does not enforce nonnegativity of $\boldsymbol{W}$. The standard approach to nonnegativity in the HALS is to apply the projection $[x]_+ = \max\{0, x\}$ onto the entries of $\underline{\boldsymbol{w}}_t$ for each t. Thus we have: $\underline{\boldsymbol{w}}_t^{(k+1)} = \left[\underline{\boldsymbol{w}}_t^{(k)}\right]_+$, where $\underline{\boldsymbol{w}}_t^{(k)}$ is calculated by (10). This simple projection used in the standard ALS algorithm does not ensure monotonic convergence. However, the projections in the HALS are nested and hierarchical. Note that the calculation of $\underline{w}_{t+1}^{(k+1)}$ involves $\underline{w}_t^{(k+1)}$, which is much more than only $\underline{w}_t^{(k)}$. Due to the nested and subsequent projections, the convergence of the HALS is monotonic and optimal according to the KKT optimality conditions [11].

After updating the whole matrix $\boldsymbol{W}$, its columns are normalized to the unit l_1 norm.

Let $\boldsymbol{A}^{(k+1)} = \boldsymbol{Y}\boldsymbol{W}^{(k+1)}$. The factor $\boldsymbol{X}$ can be estimated by solving the following regularized least squares problem with nonnegativity constraints:

$$\min_{\mathbf{X}\geq\mathbf{0}} \frac{1}{2}||\boldsymbol{Y} - \boldsymbol{A}^{(k+1)}\boldsymbol{X}||_F^2 + \lambda_X \Phi(\boldsymbol{X}), \tag{11}$$

where $\Phi(\boldsymbol{X})$ is a penalty function to enforce sparsity in $\boldsymbol{X}$, and λ_X is a penalty parameter. If the clusters are disjoint, $\boldsymbol{X}$ should be a binary matrix. For partially overlapping clusters, $\boldsymbol{X}$ should still be quite sparse.

There are many ways to enforce the sparsity in the estimated factor. Here we assume one of the most efficient and simple approach that was nearly simultaneously proposed in [16] and [17]. Let $\Phi(\boldsymbol{X}) = \mathrm{tr}\{\boldsymbol{X}^T\boldsymbol{E}_J\boldsymbol{X}\}$, where $\boldsymbol{E}_J \in \mathbb{R}_+^{J\times J}$ is a matrix of all ones. After reformulating the problem (11) according to [17], the solution $\boldsymbol{X}$ can be obtained from:

$$\min_{\mathbf{X}\geq\mathbf{0}} \left|\left| \begin{pmatrix} \boldsymbol{A}^{(k+1)} \\ \sqrt{\lambda_X}\mathbf{1}_{1\times J} \end{pmatrix} \boldsymbol{X} - \begin{pmatrix} \boldsymbol{Y} \\ \mathbf{0}_{1\times T} \end{pmatrix} \right|\right|_F^2. \tag{12}$$

To solve the system (12), we used the Fast Combinatorial Nonnegative Least Squares (FC-NNLS) algorithm [18].

3.2 β-CNMF

The model (1) can be decomposed with respect to the entries of $\boldsymbol{W}$ as:

$$\boldsymbol{Y} = \sum_{r\neq t} \boldsymbol{y}_r \underline{\boldsymbol{w}}_r \boldsymbol{X} + \sum_{s\neq j} \boldsymbol{y}_t w_{ts} \underline{\boldsymbol{x}}_s + w_{tj}\boldsymbol{y}_t\underline{\boldsymbol{x}}_j = \tilde{\boldsymbol{Y}}^{(t,j)} + w_{tj}\boldsymbol{y}_t\underline{\boldsymbol{x}}_j. \tag{13}$$

Let $\boldsymbol{Y}^{(t,j)} = \boldsymbol{Y} - \tilde{\boldsymbol{Y}}^{(t,j)}$ and $\boldsymbol{Q}^{(t,j)} = w_{tj}\boldsymbol{y}_t\underline{\boldsymbol{x}}_j$, thus $y_{in}^{(t,j)} = [\boldsymbol{Y}^{(t,j)}]_{in}$ and $q_{in}^{(t,j)} = [\boldsymbol{Q}^{(t,j)}]_{in} = w_{tj}y_{it}x_{jn}$.

Assuming that the disimilarity between the observation $y_{in}^{(t,j)}$ and the model $q_{in}^{(t,j)}$ is expressed by the β-divergence [2], we have:

$$D^{(\beta)}(y_{in}^{(t,j)}||q_{in}^{(t,j)}) = \left(y_{in}^{(t,j)}\right)^{\beta+1} \psi\left(\frac{q_{in}^{(t,j)}}{y_{in}^{(t,j)}}\right), \tag{14}$$

where $\psi(z) = \frac{1}{\beta(1+\beta)}\left(1-(\beta+1)z^{\beta}+\beta z^{\beta+1}\right)$. For $z \in \mathbb{R}_+$ and $\beta \in (0,1]$, $\psi(z)$ is strictly convex. The joint β-divergence has the form: $D^{(\beta)}(\boldsymbol{Y}^{(t,j)}||\boldsymbol{Q}^{(t,j)}) = \sum_{i=1}^{I}\sum_{n=1}^{T} D^{(\beta)}(y_{in}^{(t,j)}||q_{in}^{(t,j)})$.

From the stationarity condition $\nabla_{w_{tj}} D^{(\beta)}(\boldsymbol{Y}^{(t,j)}||\boldsymbol{Q}^{(t,j)}) \triangleq 0$, we have:

$$\begin{aligned}\nabla_{w_{tj}} D^{(\beta)}(\boldsymbol{Y}^{(t,j)}||\boldsymbol{Q}^{(t,j)}) &= \sum_{i,n}\left(q_{in}^{(t,j)} - y_{in}^{(t,j)}\right)(q_{in}^{(t,j)})^{\beta-1} y_{it} x_{jn} \\ &= \sum_{i,n}\left(w_{tj}^{\beta} y_{it}^{\beta+1} x_{jn}^{\beta+1} - y_{in}^{(t,j)} w_{tj}^{\beta-1} y_{it}^{\beta} x_{jn}^{\beta}\right) \triangleq 0. \end{aligned} \tag{15}$$

After straightforward calculations, the update rule for w_{tj} is derived from (15):

$$w_{tj} = \frac{\sum_{i,n} y_{in}^{(t,j)} y_{it}^{\beta} x_{jn}^{\beta}}{\sum_{i,n} y_{it}^{\beta+1} x_{jn}^{\beta+1}}. \tag{16}$$

Inserting $y_{in}^{(t,j)} = y_{in} - [\boldsymbol{YWX}]_{in} + w_{tj} y_{it} x_{jn}$ to (16), we obtain the simplified update rule for w_{tj}:

$$\begin{aligned} w_{tj} &\leftarrow \frac{\sum_{i,n}\left(y_{in} - [\boldsymbol{YWX}]_{in} + w_{tj} y_{it} x_{jn}\right) y_{it}^{\beta} x_{jn}^{\beta}}{\sum_{i,n} y_{it}^{\beta+1} x_{jn}^{\beta+1}} \\ &= \frac{\sum_{i,n} y_{in} y_{it}^{\beta} x_{jn}^{\beta} - \sum_{i,n} [\boldsymbol{YWX}]_{in} y_{it}^{\beta} x_{jn}^{\beta} + w_{tj}\sum_i y_{it}^{\beta+1}\sum_n x_{jn}^{\beta+1}}{\sum_{i,n} y_{it}^{\beta+1} x_{jn}^{\beta+1}} \\ &= w_{tj} + \frac{[(\boldsymbol{Y}^{\beta})^T \boldsymbol{Y}(\boldsymbol{X}^{\beta})^T]_{tj} - [(\boldsymbol{Y}^{\beta})^T \boldsymbol{YWX}(\boldsymbol{X}^{\beta})^T]_{tj}}{(\boldsymbol{1}_I^T \boldsymbol{Y}^{\beta+1})_t (\boldsymbol{X}^{\beta+1}\boldsymbol{1}_T)_j}. \end{aligned} \tag{17}$$

The operator $\boldsymbol{Z}^{\beta} = [z_{ij}^{\beta}]$ means element-wise raise to power β. The update rule (17) can be parallelized with respect to the index t or j (but not jointly). For $T >> J$, higher efficiency can be obtained if only one loop **for** (sweeping through t) is used. Thus:

$$w_{t,*}^{(k+1)} = w_{t,*}^{(k)} + \frac{\left[(\boldsymbol{Y}^{\beta})^T \boldsymbol{Y}(\boldsymbol{X}^{\beta})^T\right]_{t,*} - \left[(\boldsymbol{Y}^{\beta})^T \boldsymbol{Y}\right]_{t,*} \tilde{\boldsymbol{W}}^{(k)} \boldsymbol{X}(\boldsymbol{X}^{\beta})^T}{(\boldsymbol{1}_I^T \boldsymbol{Y}^{\beta+1})_t (\boldsymbol{X}^{\beta+1}\boldsymbol{1}_T)_*}, \tag{18}$$

where $\tilde{\boldsymbol{W}}^{(k)} = [\underline{\boldsymbol{w}}_1^{(k+1)}; \ldots; \underline{\boldsymbol{w}}_{t-1}^{(k+1)}; \underline{\boldsymbol{w}}_t^{(k)}; \ldots; \underline{\boldsymbol{w}}_T^{(k)}] \in \mathbb{R}^{T\times J}$.

Neglecting the computational complexity for raising to power β, the matrices $(\boldsymbol{Y}^{\beta})^T\boldsymbol{Y}$, $\boldsymbol{X}(\boldsymbol{X}^{\beta})^T$ and $(\boldsymbol{Y}^{\beta})^T\boldsymbol{Y}(\boldsymbol{X}^{\beta})^T$ can be precomputed with the costs:

$O(IT^2)$, $O(TJ^2)$ and $O(JT^2) + O(IT^2)$, respectively. Hence the overall computational complexity for k iterations with the update rule (18) can be roughly estimated as $O(IT^2 + JT^2 + TJ^2 + kT^2J^2)$. Assuming $J << \min\{I, T\}$, we have: $O(IT^2 + kT^2J^2)$. It is therefore J higher than for the HALS-based CNMF.

If $T >> J$, i.e. the clusters are assumed to include many samples, the centroids do not have to be calculated using all samples. To accelerate the computations both for the HALS-CNMF and β-CNMF, the update rules in (10) and (18) may be applied to only the selected rows of $\boldsymbol{W}$ in each iterative step. The selection can be random, and the number of the selected rows should depend on the rate T/J. In the experiments, we select only 10 percent of the rows in each iteration.

4 Experiments

The proposed algorithms were tested for solving partitional clustering problems using various datasets that are briefly characterized in Table 1.

Table 1. Details of the datasets

Datasets	Variables (I)	Samples (T)	Classes (J)	Sparsity [%]
Gaussian mixture	3	3000	3	0
Hand-written digits	64	5620	10	3.1
TPD	8190	888	6	98.45
Reuters	6191	2500	10	99.42

The samples in the *Gaussian mixture* dataset are generated randomly from a mixture of three 3D Gaussian distributions with the following parameters:

$$\boldsymbol{\mu}_1 = [40, 80, -30]^T, \quad \boldsymbol{\mu}_2 = [70, -40, 60]^T, \quad \boldsymbol{\mu}_3 = [20, 20, 30]^T,$$

$$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 50 & -0.2 & 0.1 \\ -0.2 & 0.1 & 0.1 \\ 0.1 & 0.1 & 5 \end{bmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{bmatrix} 50 & -5 & -1 \\ -5 & 5 & -0.5 \\ -1 & 0.5 & 1 \end{bmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 5 \end{bmatrix}.$$

Obviously, all the covariance matrices are positive-definite. From each distribution 500 samples are generated, hence $\mathbf{Y} \in \mathbb{R}^{3\times 1500}$.

The dataset entitled *Hand-written digits* is taken from the UCI Machine Learning Repository [19]. It contains hand-written digits used for optical recognition.

The datasets *TPD* and *Reuters* contain textual documents that should be grouped according to their semantic similarity. The documents in the first one come from the *TopicPlanet* document collection. We selected 888 documents classified into 6 topics: *air-travel, broadband, cruises, domain-names, investments, technologies*, which gives 8190 words after having been parsed. Thus $\mathbf{Y} \in \mathbb{R}^{8190\times 888}$ and $J = 6$. The documents in the *Reuters* database belong to the following topics: *acq, coffee, crude, eran, gold, interest, money-fx, ship, sugar, trade*. We selected 2500 documents that have 6191 distinctive and meaningful words; thus $\mathbf{Y} \in \mathbb{R}^{6191\times 2500}$ and $J = 10$. Both datasets are very sparse, since each document contains only a small portion of the words from the dictionary.

Several NMF algorithms are compared with respect to the efficiency for solving clustering problems. The proposed algorithms are referred to as the HALS-CNMF and β-CNMF. The other algorithms are listed as follows: HALS [8], UO-NMF(A) (Uni-orth. NMF with orthogonalization of the feature matrix) [20], UO-NMF(X) (Uni-orth. NMF with orthogonalization of the encoding matrix) [20], Bio-NMF (Bi-orthogonal NMF) [20], Cx-NMF (standard multiplicative convex NMF) [3], and k-means (standard Matlab implementation for minimization of the Euclidean distance). In the β-CNMF, we set $\beta = 5$.

All the tested algorithms were initialized by the same random initializer generated from an uniform distribution. To analyze the efficiency of the discussed methods, 100 Monte Carlo (MC) runs of each algorithm were carried out, each time the initial matrices were different. All the algorithms were implemented using the same computational strategy, i.e. the same stopping criteria are applied to all the algorithms, and the maximum number of inner iterations for updating the factor $\boldsymbol{A}$, $\boldsymbol{W}$ or $\boldsymbol{X}$ is set to 10.

The quality of clustering is evaluated with the Purity measure [20] that reflects the accuracy of clustering. Fig. 1 shows the statistics of the Purity obtained from 100 MC runs of the tested algorithms. The average runtime is given in Table 2.

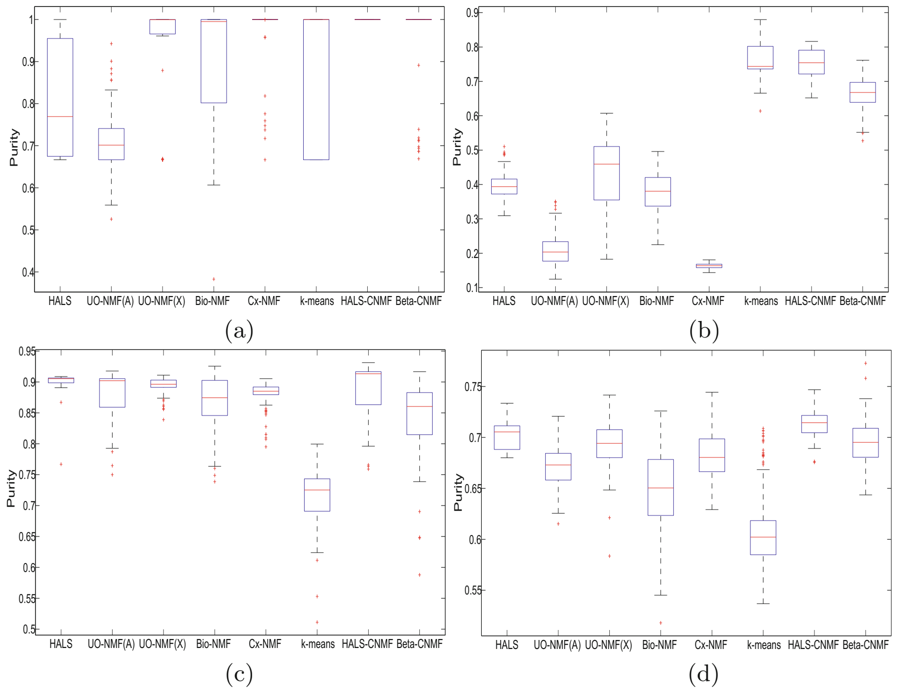

Fig. 1. Statistics of the purity measure for clustering the following datasets: (a) Gaussian mixture; (b) Hand-written digits; (c) TPD; (d) Reuters

Table 2. Average runtime [in seconds] of the tested algorithms: 1 – HALS, 2 – UO-NMF(A), 3 – UO-NMF(X), 4 – Bio-NMF, 5 – Cx-NMF, 6 – k-means, 7 – HALS-CNMF, 8 – β-CNMF

Datasets	1	2	3	4	5	6	7	8
Gaussian mixture	0.077	0.079	0.16	0.27	72.9	0.0053	2.76	4.66
Hand-written digits	1.02	0.71	1.1	2.26	39.3	0.53	25.8	24.5
TPD	3.85	2.26	4.34	6.07	14.1	36.53	7.0	11.37
Reuters	10.12	5.41	12.4	14.2	89.75	194.6	29.7	48.2

5 Conclusions

In this paper, we proposed two versions of CNMF for clustering mixed-sign and unnecessarily sparse data points. Both algorithms are more efficient with respect to the clustering accuracy and the computational time than the standard multiplicative CNMF. The results presented in Fig. 1 show that the HALS-CNMF gives the best clustering accuracy for the analyzed datasets. The β-CNMF can be tuned to the distribution of data points with the parameter β. If the number of variables in the dataset is much larger than the number of clusters, both proposed CNMF algorithms are faster than the k-means (see Table 2). When the number of samples is very large, the proposed algorithms provide high accuracy of clustering but at the cost of an increased computational cost.

Summing up, the proposed CNMF algorithms seem to be efficient for clustering mixed-signed data points. They can be also combined with the CH-NMF for clustering big data.

References

1. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature 401, 788–791 (1999)
2. Cichocki, A., Zdunek, R., Phan, A.H., Amari, S.I.: Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation. Wiley and Sons (2009)
3. Ding, C., Li, T., Jordan, M.I.: Convex and semi-nonnegative matrix factorizations. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(1), 45–55 (2010)
4. Thurau, C., Kersting, K., Bauckhage, C.: Convex non-negative matrix factorization in the wild. In: Proc. The 2009 Ninth IEEE International Conference on Data Mining, ICDM 2009, pp. 523–532. IEEE Computer Society, Washington, DC (2009)
5. Esser, E., Möller, M., Osher, S., Sapiro, G., Xin, J.: A convex model for nonnegative matrix factorization and dimensionality reduction on physical space. IEEE Transactions on Image Processing 21(7), 3239–3252 (2012)
6. Krishnamurthy, V., d'Aspremont, A.: Convex algorithms for nonnegative matrix factorization (2012), http://arxiv.org/abs/1207.0318
7. Cichocki, A., Zdunek, R., Amari, S.-I.: Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization. In: Davies, M.E., James, C.J., Abdallah, S.A., Plumbley, M.D. (eds.) ICA 2007. LNCS, vol. 4666, pp. 169–176. Springer, Heidelberg (2007)

8. Cichocki, A., Phan, A.H.: Fast local algorithms for large scale nonnegative matrix and tensor factorizations. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E92-A(3), 708–721 (2009)
9. Han, L., Neumann, M., Prasad, U.: Alternating projected Barzilai-Borwein methods for nonnegative matrix factorization. Electronic Transactions on Numerical Analysis 36, 54–82 (2009-2010)
10. Kim, J., Park, H.: Fast nonnegative matrix factorization: An active-set-like method and comparisons. SIAM J. Sci. Comput. 33(6), 3261–3281 (2011)
11. Gillis, N., Glineur, F.: Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization. Neural Comput. 24(4), 1085–1105 (2012)
12. Chen, W., Guillaume, M.: HALS-based NMF with flexible constraints for hyperspectral unmixing. EURASIP J. Adv. Sig. Proc. 54, 1–14 (2012)
13. Zdunek, R.: Nonnegative Matrix and Tensor Factorization: Applications to Classification and Signal Processing. Publishing House of Wroclaw University of Technology, Wroclaw (2014) (in Polish).
14. Zdunek, R.: Data clustering with semi-binary nonnegative matrix factorization. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 705–716. Springer, Heidelberg (2008)
15. Yang, Z., Oja, E.: Linear and nonlinear projective nonnegative matrix factorization. IEEE Transactions Neural Networks 21(5), 734–749 (2010)
16. Zdunek, R., Cichocki, A.: Nonnegative matrix factorization with constrained second-order optimization. Signal Processing 87, 1904–1916 (2007)
17. Kim, H., Park, H.: Non-negative matrix factorization based on alternating non-negativity constrained least squares and active set method. SIAM Journal in Matrix Analysis and Applications 30(2), 713–730 (2008)
18. Benthem, M.H.V., Keenan, M.R.: Fast algorithm for the solution of large-scale non-negativity-constrained least squares problems. Journal of Chemometrics 18, 441–450 (2004)
19. Bache, K., Lichman, M.: UCI machine learning repository (2013)
20. Ding, C., Li, T., Peng, W., Park, H.: Orthogonal nonnegative matrix tri-factorizations for clustering. In: KDD 2006: Proc. of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 126–135. ACM Press, New York (2006)