GNMF with Newton-Based Methods

Rafał Zdunek[1], Anh-Huy Phan[2], and Andrzej Cichocki[2,3,4]

[1] Department of Electronics, Wroclaw University of Technology,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland
rafal.zdunek@pwr.wroc.pl
[2] Laboratory for Advanced Brain Signal Processing
RIKEN BSI, Wako-shi, Japan
[3] Warsaw University of Technology, Poland
[4] Systems Research Institute, Polish Academy of Science (PAN), Poland

Abstract. Several variants of Nonnegative Matrix Factorization (NMF) have been proposed for supervised classification of various objects. Graph regularized NMF (GNMF) incorporates the information on the data geometric structure to the training process, which considerably improves the classification results. However, the multiplicative algorithms used for updating the underlying factors may result in a slow convergence of the training process. To tackle this problem, we propose to use the Spectral Projected Gradient (SPG) method that is based on quasi-Newton methods. The results are presented for image classification problems.

Keywords: NMF, Graph-regularized NMF, SPG, Image classification.

1 Introduction

Nonnegative Matrix Factorization (NMF) [1] decomposes a nonnegative matrix into lower-rank factor matrices that have nonnegative entries and usually some physical meaning. When NMF is applied to the matrix of training samples, we obtain sparse nonnegative feature vectors and coefficients of their nonnegative combination. The vectors of the coefficients lie in a low-dimensional latent component space. Hence, NMF is often regarded as a dimensionality reduction technique, and it has been widely applied for classification of various objects [2–6].

As reported in [7], the factor matrices obtained with NMF are generally nonunique. Several attempts have been done to additionally constrain them to satisfy a certain degree of sparsity, smoothness, uncorrelatedness, or orthogonality [2]. Cai *et al.* [8, 9] noticed that the projection from the high-dimensional observation space to the low-dimensional space should preserve the data geometrical structure. That is, any training samples forming one class should, after being projected, belong to the same class in the latent component space. Thus, they proposed Graph regularized NMF (GNMF) [8] that constrains one of the factor matrices with the information on the data geometric structure encoded in the nearest-neighbor graph of the training samples. This constraint was imposed to NMF by a specifically designed regularization term in the objective function

V. Mladenov et al. (Eds.): ICANN 2013, LNCS 8131, pp. 90–97, 2013.

that was then minimized with the standard multiplicative algorithm [2]. Guan *et al.* [10] considerably accelerated the convergence of GNMF by using additive gradient descent updates.

In this paper, we propose to improve the convergence rate of GNMF updates even more, by applying another Newton-based methods that provide the estimates according to the Karush-Kuhn-Tucker (KKT) optimality conditions. First, we formulate the Quadratic Programming (QP) problems for minimizing the penalized objective function. The QP problems can be efficiently solved with many numerical algorithms. To tackle large-scale classification problems, we suggest to use the modified Spectral Projected Gradient (SPG) method that belongs to the class of quasi-Newton methods. Moreover, we also propose to control the penalty parameters iteratively by some schedule included in the alternating update scheme.

The paper is organized in the following way. The next section discusses the Graph-regularized NMF. Section 3 is concerned with the optimization algorithms. The numerical experiments for image classification problems are presented in Section 4. Finally, the conclusions are drawn in Section 5.

2 Graph-Regularized NMF

Let $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T] \in \mathbb{R}_+^{I \times T}$, where $\boldsymbol{y}_t \in \mathbb{R}_+^I$ is the t-th training sample. Applying NMF to $\boldsymbol{Y}$, we get $\boldsymbol{Y} \cong \boldsymbol{A}\boldsymbol{X}$, where the columns of the matrix $\boldsymbol{A} \in \mathbb{R}_+^{I \times J}$ represent the feature vectors, and the columns of the matrix $\boldsymbol{X} \in \mathbb{R}_+^{J \times T}$ are encoding vectors.

In several variants of NMF, the objective function can be expressed by the quadratic function:

$$\Psi(\boldsymbol{A}, \boldsymbol{X}) = \frac{1}{2}||\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{X}||_F^2 + \frac{\alpha_X}{2}\operatorname{tr}(\boldsymbol{X}\boldsymbol{L}_X\boldsymbol{X}^T) + \frac{\alpha_A}{2}\operatorname{tr}(\boldsymbol{A}^T\boldsymbol{L}_A\boldsymbol{A}), \tag{1}$$

where $\boldsymbol{L}_X \in \mathbb{R}^{T \times T}$ and $\boldsymbol{L}_A \in \mathbb{R}^{I \times I}$ are symmetric weighting matrices. In supervised classification, the matrix $\boldsymbol{L}_X$ contains the information on assignments of the training samples to their classes. In DNMF [4], it is determined by the matrix of inner- and outer-class scattering. In GNMF [8], $\boldsymbol{L}_X$ is the graph Laplacian matrix that represents a data geometrical structure in the observation space. It takes form: $\boldsymbol{L}_X = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{W} = [w_{nm}] \in \mathbb{R}_+^{T \times T}$ contains the entries that determine the edges in the nearest neighbor graph of the observed points, and $\boldsymbol{D} = \operatorname{diag}\left(\sum_{m=1}^T w_{nm}\right) \in \mathbb{R}_+^{T \times T}$. The edges can be determined by the hard connections:

$$w_{nm} = \begin{cases} 1, \text{ if } \quad \boldsymbol{y}_n \in \mathcal{N}_p(\boldsymbol{y}_m), \text{ or } \boldsymbol{y}_m \in \mathcal{N}_p(\boldsymbol{y}_n), \\ 0, \qquad\qquad \text{otherwise} \end{cases} \tag{2}$$

where $\mathcal{N}_p(\boldsymbol{y}_t)$ is the p nearest neighbor of the sample $\boldsymbol{y}_t$. We can also use the Heat kernel weighting:

$$w_{nm} = \begin{cases} \exp\left\{-\frac{||\boldsymbol{y}_n - \boldsymbol{y}_m||_2^2}{\sigma}\right\}, \text{ if } \quad \boldsymbol{y}_n \in \mathcal{N}_p(\boldsymbol{y}_m), \text{ or } \boldsymbol{y}_m \in \mathcal{N}_p(\boldsymbol{y}_n), \\ 0, \qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{3}$$

or the cosine measure:

$$w_{nm} = \begin{cases} \boldsymbol{y}_n^T \boldsymbol{y}_m, \text{ if } \quad \boldsymbol{y}_n \in \mathcal{N}_p(\boldsymbol{y}_m), \text{ or } \boldsymbol{y}_m \in \mathcal{N}_p(\boldsymbol{y}_n), \\ 0, \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{4}$$

The matrix $\boldsymbol{L}_A$ in (1) can enforce the smoothness in the feature vectors (the column vectors in $\boldsymbol{A}$) or other modality. We assumed the simplest approach to the smoothness by setting $\boldsymbol{L}_A = \boldsymbol{I}_I$, where $\boldsymbol{I}_I \in \mathbb{R}_+^{I \times I}$ is an identity matrix.

3 Algorithm

Since the matrix $\boldsymbol{L}_X$ in (1) is a symmetric and positive definite, the regularization term $\text{tr}(\boldsymbol{X}\boldsymbol{L}_X\boldsymbol{X}^T)$ can be reformulated as follows:

$$\Psi_r(\boldsymbol{X}) = \text{tr}(\boldsymbol{X}\boldsymbol{L}_X\boldsymbol{X}^T) = ||\boldsymbol{X}\boldsymbol{L}_X^{\frac{1}{2}}||_F^2 = ||(\boldsymbol{L}_X^{\frac{1}{2}} \otimes \boldsymbol{I}_J)\boldsymbol{x}||_2^2 = \boldsymbol{x}^T(\boldsymbol{L}_X \otimes \boldsymbol{I}_J)\boldsymbol{x}, \tag{5}$$

where $\boldsymbol{x} = \text{vec}(\boldsymbol{X}) \in \mathbb{R}^{JT}$ is a vectorized form of $\boldsymbol{X}$, and $\otimes$ stands for the Kronecker product.

Considering the function (5), the minimization problem: $\min_{\mathbf{X}} \Psi(\boldsymbol{A}, \boldsymbol{X})$, s.t. $\boldsymbol{X} \geq \mathbf{0}$ can be expressed in terms of the Quadratic Programming (QP) problem: $\min_{\boldsymbol{x}} \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}_X\boldsymbol{x} + \boldsymbol{c}_X^T\boldsymbol{x}$, s.t. $\boldsymbol{x} \geq 0$, where $\boldsymbol{Q}_X = \boldsymbol{I}_T \otimes \boldsymbol{A}^T\boldsymbol{A} + \alpha_X\boldsymbol{L}_X \otimes \boldsymbol{I}_J \in \mathbb{R}^{JT \times JT}$ and $\boldsymbol{c}_X = -\text{vec}(\boldsymbol{A}^T\boldsymbol{Y}) \in \mathbb{R}^{JT}$.

Similarly, the matrix $\boldsymbol{A}$ can be also computed by formulating the QP problem: $\min_{\boldsymbol{a}} \frac{1}{2}\boldsymbol{a}^T\boldsymbol{Q}_A\boldsymbol{a} + \boldsymbol{c}_A^T\boldsymbol{a}$, s.t. $\boldsymbol{a} \geq 0$, where $\boldsymbol{a} = \text{vec}(\boldsymbol{A}^T) \in \mathbb{R}^{IJ}$, $\boldsymbol{Q}_A = (\boldsymbol{X}\boldsymbol{X}^T + \alpha_A\boldsymbol{I}_J) \otimes \boldsymbol{I}_I \in \mathbb{R}^{IJ \times IJ}$ and $\boldsymbol{c}_A = -\text{vec}(\boldsymbol{Y}\boldsymbol{X}^T) \in \mathbb{R}^{IJ}$.

Since the function (1) is quadratic with respect to both arguments $\boldsymbol{A}$ and $\boldsymbol{X}$ (but not jointly), the matrices $\boldsymbol{Q}_A$ and $\boldsymbol{Q}_X$ are equivalent to the Hessian matrices for $\boldsymbol{A}$ and $\boldsymbol{X}$, respectively. When $\alpha_A > 0$, the matrix $\boldsymbol{Q}_A$ is positive definite. Under the assumption of positive definiteness of the matrix $\boldsymbol{L}_X$, the matrix $\boldsymbol{Q}_X$ is also positive definite. Hence, both QP problems are strictly convex. To solve such problems, we can use many numerical algorithms such as the Active-Set (AS), Interior-Point (IP), and Spectral Projected Gradient (SPG) [11]. These algorithm are based on the Newton or quasi-Newton updates.

Note that the matrix $\boldsymbol{Q}_A$ has a block-diagonal structure, and hence the updates of $\boldsymbol{A}$ might be considerably accelerated by transforming the nonnegative least-squares problem: $\min_{\mathbf{A} \geq 0} \frac{1}{2}||\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{X}||_F^2 + \frac{\alpha_A}{2}||\boldsymbol{A}||_F^2$ to the normal equations $\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{A}^T = \boldsymbol{X}\boldsymbol{Y}^T$ subject to the nonnegativity constraints $\boldsymbol{A} \geq \mathbf{0}$. Then, the solution can be efficiently searched with the FC-NNLS algorithm that was proposed by Benthem and Keenan [12], and then adapted to NMF problems in [13].

The updates for $\boldsymbol{X}$ cannot be accelerated in the similar way, however, there is still a possibility of applying some quasi-Newton method without formulating the Hessian $\boldsymbol{Q}_X$. Note that the matrix $\boldsymbol{Q}_X$ is very large when the number of training samples is large, and it is rather a dense matrix due to the matrix $\boldsymbol{L}_X$. One of these possibilities is to use the SPG method [14] that combines the standard gradient projection scheme with the nonmonotonic Barzilai-Borwein

(BB) method [11]. It is used for minimization of convex functions subject to box-constraints.

In the SPG method, the descent direction $\boldsymbol{p}_t^{(k)}$ for updating the vector $\boldsymbol{x}_t$ in the k-th iteration is defined as follows:

$$\boldsymbol{p}_t^{(k)} = \left[\boldsymbol{x}_t^{(k)} - (\alpha_t^{(k)})^{-1}\nabla_{\boldsymbol{x}_t}\Psi(\boldsymbol{A}, \boldsymbol{x}_t^{(k)})\right]_+ - \boldsymbol{x}_t^{(k)}, \tag{6}$$

for $\alpha_t^{(k)} > 0$ selected in such a way that the matrix $\alpha_t^{(k)}\boldsymbol{I}_J$ approximates the Hessian matrix.

In [2], this method was adopted to parallel processing of all column vectors in $\boldsymbol{X}$. Using this approach, we have the update rule:

$$\boldsymbol{X}^{(k+1)} = \boldsymbol{X}^{(k)} + \boldsymbol{P}^{(k)}\boldsymbol{Z}^{(k)}, \tag{7}$$

where $\boldsymbol{Z}^{(k)} = \operatorname{diag}\{\boldsymbol{\eta}^{(k)}\}$. The column vectors of $\boldsymbol{P}^{(k)} \in \mathbb{R}^{J\times T}$ and the entries of the vector $\boldsymbol{\eta}^{(k)} \in \mathbb{R}_+^T$ are descent directions and steplengths for updating the vectors $\{\boldsymbol{x}_t\}$, respectively. According to (6), the matrix $\boldsymbol{P}^{(k)}$ has the form:

$$\boldsymbol{P}^{(k)} = \left[\boldsymbol{X}^{(k)} - \boldsymbol{G}_X^{(k)}\boldsymbol{D}^{(k)}\right]_+ - \boldsymbol{X}^{(k)}, \tag{8}$$

where $\boldsymbol{G}_X^{(k)} = \nabla_{\mathbf{X}}\Psi(\boldsymbol{A}, \boldsymbol{X}^{(k)}) \in \mathbb{R}^{J\times T}$ and $\boldsymbol{D}^{(k)} = \operatorname{diag}\{(\alpha_t^{(k)})^{-1}\} \in \mathbb{R}^{T\times T}$.

The coefficients $\{\alpha_t^{(k)}\}$ can be obtained from the secant equation that is given by $\boldsymbol{S}^{(k)}\operatorname{diag}\{\alpha_t^{(k+1)}\} = \boldsymbol{W}^{(k)}$, where $\boldsymbol{S}^{(k)} = \boldsymbol{X}^{(k+1)} - \boldsymbol{X}^{(k)}$ and $\boldsymbol{W}^{(k)} = \nabla_{\mathbf{X}}\Psi(\boldsymbol{A}, \boldsymbol{X}^{(k+1)}) - \nabla_{\mathbf{X}}\Psi(\boldsymbol{X}^{(k)})$. For the minimization of the objective function (1) with respect to $\boldsymbol{X}$, the matrix $\boldsymbol{W}^{(k)}$ takes the form: $\boldsymbol{W}^{(k)} = \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{S}^{(k)} + \alpha_X\boldsymbol{S}^{(k)}\boldsymbol{L}_X$. From (7) we have: $\boldsymbol{S}^{(k)} = \boldsymbol{P}^{(k)}\boldsymbol{Z}^{(k)}$. In consequence, the secant equation leads to:

$$\begin{aligned}
\boldsymbol{\alpha}^{(k+1)} &= \frac{\operatorname{diag}\left\{(\boldsymbol{S}^{(k)})^T\boldsymbol{W}^{(k)}\right\}}{\operatorname{diag}\left\{(\boldsymbol{S}^{(k)})^T\boldsymbol{S}^{(k)}\right\}} = \frac{\operatorname{diag}\left\{(\boldsymbol{S}^{(k)})^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{S}^{(k)} + \alpha_X(\boldsymbol{S}^{(k)})^T\boldsymbol{S}^{(k)}\boldsymbol{L}_X\right\}}{\operatorname{diag}\left\{(\boldsymbol{S}^{(k)})^T\boldsymbol{S}^{(k)}\right\}} \\
&= \frac{\operatorname{diag}\left\{(\boldsymbol{P}^{(k)})^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{P}^{(k)} + \alpha_X(\boldsymbol{P}^{(k)})^T\boldsymbol{P}^{(k)}\boldsymbol{Z}^{(k)}\boldsymbol{L}_X(\boldsymbol{Z}^{(k)})^{-1}\right\}}{\operatorname{diag}\left\{(\boldsymbol{P}^{(k)})^T\boldsymbol{P}^{(k)}\right\}} \\
&= \frac{\mathbf{1}_J^T\left[\boldsymbol{P}^{(k)} \circledast \left(\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{P}^{(k)} + \alpha_X\boldsymbol{P}^{(k)}\boldsymbol{Z}^{(k)}\boldsymbol{L}_X(\boldsymbol{Z}^{(k)})^{-1}\right)\right]}{\mathbf{1}_J^T\left[\boldsymbol{P}^{(k)} \circledast \boldsymbol{P}^{(k)}\right]},
\end{aligned} \tag{9}$$

where $\circledast$ stands for the Hadamard product, and the operation $\operatorname{diag}\{\boldsymbol{M}\}$ creates a vector containing the main diagonal entries of a matrix $\boldsymbol{M}$. Note that the matrix $\boldsymbol{Z}^{(k)}$ is diagonal, so the product $\boldsymbol{Z}^{(k)}\boldsymbol{L}_X(\boldsymbol{Z}^{(k)})^{-1}$ can be readily calculated.

The steplengths can be estimated by solving the minimization problem:

$$\boldsymbol{\eta}_*^{(k)} = \arg\min_{\boldsymbol{\eta}^{(k)}} \Psi\left(\boldsymbol{A}, \boldsymbol{X}^{(k)} + \boldsymbol{P}^{(k)}\operatorname{diag}\{\boldsymbol{\eta}^{(k)}\}\right). \tag{10}$$

If $\alpha_X = 0$, the problem (10) can be expressed in a closed-form. Otherwise, iterative updates must be used, e.g. the Armijo rule [11].

The final form of the modified SPG algorithm is given by Algorithm 1. The final form of the NMF algorithm used in the training process is given by Algorithm 2.

Algorithm 1. SPG algorithm

Input : $\boldsymbol{Y} \in \mathbb{R}_+^{I \times T}$, $\boldsymbol{A} \in \mathbb{R}_+^{I \times J}$, $\boldsymbol{X}^{(0)} \in \mathbb{R}_+^{J \times T}$ - initial guess, k_{max} - number of iterations for SPG updates, $\alpha_{min} > 0$, $\alpha_{max} > 0$, $\forall t : \bar{\alpha}_t^{(0)} = \frac{1}{2}\alpha_{max}$,

Output: $\hat{\boldsymbol{X}}$ - estimated factor matrices,

for $k = 0, 1, \ldots, k_{max}$ **do**

$\boldsymbol{G}_X^{(k)} = \nabla_{\mathbf{X}} \Psi(\boldsymbol{A}, \boldsymbol{X}^{(k)}) = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{X}^{(k)} - \boldsymbol{Y}) + \alpha_X \boldsymbol{X}^{(k)} \boldsymbol{L}_X$; `// Gradient`

$\boldsymbol{P}^{(k)} = \left[\boldsymbol{X}^{(k)} - \boldsymbol{G}_X^{(k)} \operatorname{diag}\{(\bar{\alpha}_t^{(k)})^{-1}\}\right]_+ - \boldsymbol{X}^{(k)}$; `// Descent direction`

$\bar{\boldsymbol{\eta}}^{(k)} = \max\{0, \min\{1, \boldsymbol{\eta}^{(k)}\}\}$; `// where` $\boldsymbol{\eta}^{(k)}$ `is estimated with (10)`

$\boldsymbol{X}^{(k+1)} = \boldsymbol{X}^{(k)} + \boldsymbol{P}^{(k)} \operatorname{diag}\{\bar{\boldsymbol{\eta}}^{(k)}\}$;

$\bar{\boldsymbol{\alpha}}^{(k+1)} = \max\{\alpha_{min}, \min\{\alpha_{max}, \boldsymbol{\alpha}^{(k+1)}\}\}$; `// where` $\boldsymbol{\alpha}^{(k+1)}$ `is set to (9)`

In the training process, we obtain the nonnegative matrices $\boldsymbol{A}$ and $\boldsymbol{X}$. The column vectors of $\boldsymbol{X}$ contain the discriminant information. To classify the test sample $\tilde{\boldsymbol{y}}$, first we need to project it onto the subspace spanned by the column vectors of the matrix $\boldsymbol{A}$. As a result, we obtain $\tilde{\boldsymbol{x}} \in \mathbb{R}_+^J$. This step can be carried out with the SPG, assuming $\alpha_X = 0$. Then, the following problem is solved: $t_* = \arg\min_{1 \le t \le T} ||\tilde{\boldsymbol{x}} - \boldsymbol{x}_t||_2$, which gives the index t_* of the class to which the sample $\tilde{\boldsymbol{y}}$ is classified.

4 Experiments

The experiments are carried out for classification of facial images taken from the ORL database[1]. It contains 400 frontal facial images of 40 people (10 pictures per person). We selected 8 training images randomly from each class, and the remaining 2 images are used for testing.

We test the following NMF algorithms: MUE (standard multiplicative Lee-Seung algorithm for the Euclidean distance) [1], GNMF [8], MD-NMF [10], standard projected ALS [2], LPG (Lin's Projected Gradient) [15], IP (Interior-Point NMF) [16], regularized FC-NNLS [13], SPG-NMF (Algorithm 2). For the SPG algorithm, we found the optimal parameters: $\alpha_X = 10^{-5}$, $\bar{\alpha} = 10^{-12}$, $\alpha_0 = 0.01$, and $k_{max} = \min\{k, 50\}$, where k is the alternating step in Algorithm 2. The matrix $\boldsymbol{L}_X$ is determined using the hard connection criterion given by (2). The iterative process is terminated after 50 alternating steps.

[1] `http://people.cs.uchicago.edu/~dinoj/vis/orl/`

Algorithm 2. SPG-NMF Algorithm

Input : $\boldsymbol{Y} \in \mathbb{R}^{I \times T}$, J - lower rank, α_0 - initial regularization parameter,
Output: Factor matrices: $\boldsymbol{A} \in \mathbb{R}_+^{I \times J}$ and $\boldsymbol{X} \in \mathbb{R}_+^{J \times T}$

Initialize: $\boldsymbol{A}$ and $\boldsymbol{X}$ with nonnegative random numbers;
Replace negative entries (if any) in $\boldsymbol{Y}$ with zero-value, $k = 0$;
repeat
  $\alpha_A^{(k)} = \max\left\{\bar{\alpha}, 2^{-k}\alpha_0\right\}$; `// Regularization parameter schedule`
  $\boldsymbol{X}^{(k+1)} = \texttt{SPG}(\boldsymbol{Y}, \boldsymbol{A}^{(k)}, \boldsymbol{X}^{(k)}, \alpha_X)$;
  $\bar{d}_j^{(k+1)} = \sum_{t=1}^{T} x_{jt}^{(k+1)}$,
  $\boldsymbol{X}^{(k+1)} \leftarrow \text{diag}\left\{\left(\bar{d}_j^{(k+1)}\right)^{-1}\right\}\boldsymbol{X}^{(k+1)}, \quad \boldsymbol{A}^{(k)} \leftarrow \boldsymbol{A}^{(k)}\text{diag}\left\{\bar{d}_j^{(k+1)}\right\}$;
  $\bar{\boldsymbol{A}}^{(k+1)} = \texttt{FCNNLS}(\boldsymbol{Y}^T, (\boldsymbol{X}^{(k+1)})^T, (\boldsymbol{A}^{(k)})^T, \alpha_A^{(k)})$;
  $\boldsymbol{A}^{(k+1)} = (\bar{\boldsymbol{A}}^{(k+1)})^T$;
  $\bar{\bar{d}}_j^{(k+1)} = \sum_{i=1}^{I} a_{ij}^{(k+1)}$,
  $\boldsymbol{X}^{(k+1)} \leftarrow \text{diag}\left\{\bar{\bar{d}}_j^{(k+1)}\right\}\boldsymbol{X}^{(k+1)}, \quad \boldsymbol{A}^{(k+1)} \leftarrow \boldsymbol{A}^{(k+1)}\text{diag}\left\{\left(\bar{\bar{d}}_j^{(k+1)}\right)^{-1}\right\}$;
  $k \leftarrow k + 1$;
until `Stop criterion` is satisfied;

The NMF algorithms are initialized with uniformly distributed random matrices, and tested for various values of the related parameters. Fig. 1 presents the mean recognition rate versus the number of components (parameter J) obtained with different NMF algorithms.

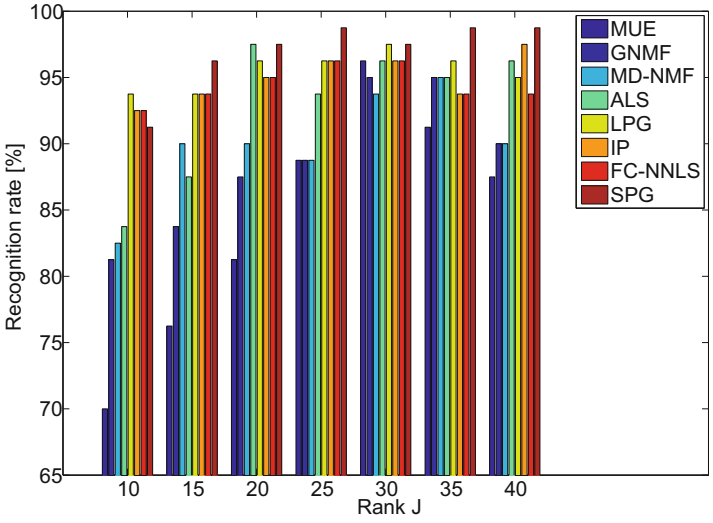

Fig. 1. Recognition rate obtained using various NMF algorithms versus the number of components J

The normalized residual errors versus the number of iterations for the selected NMF algorithms are plotted in Fig. 2.

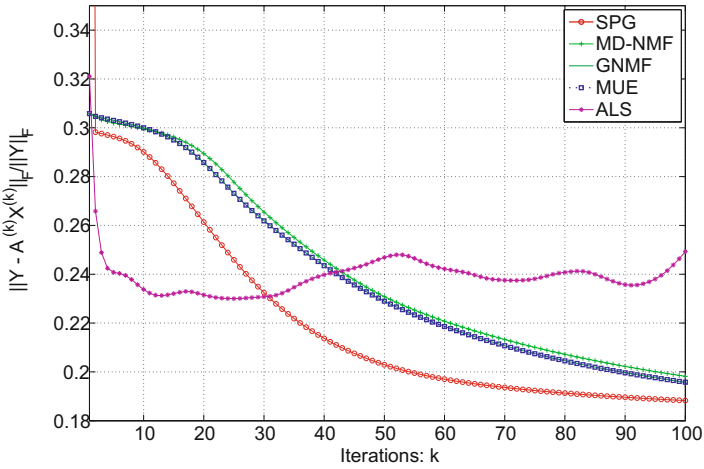

Fig. 2. Normalized residual errors versus alternating iterations

5 Conclusions

The results presented in Fig. 1 demonstrate that the SPG-NMF algorithm outperforms the other tested algorithms in terms of the recognition rate for $J > 10$. Usually an increase in the factorization rank leads to a higher recognition rate. The experiments also confirm that the NMF algorithms based on Newton-like methods (SPG, IP, FC-NNLS and ALS) converge faster than the multiplicative algorithms. This can be observed in Fig. 2 where the SPG-NMF algorithm demonstrates a better convergence behavior than the others. Initially the projected ALS algorithm converges faster but it does not guarantee a monotonic convergence. As observed in Fig. 2 the residual error of the SPG-NMF decreases monotonically with alternating steps. This behavior is also justified by the fact that both SPG and FC-NNLS algorithms converge to the solution optimal according to the KKT conditions. Moreover, the convergence of the SPG-NMF is faster than for the multiplicative algorithms since the SPG is a quasi-Newton method, i.e., the gradient direction is scaled using the information on the Hessian approximation.

Summing up, the experiments showed that the proposed algorithm works very efficiently for the facial classification problem. The usefulness of the proposed algorithm in other applications of NMF will be analyzed in the further research.

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. Qin, L., Zheng, Q., Jiang, S., Huang, Q., Gao, W.: Unsupervised texture classification: Automatically discover and classify texture patterns. Image and Vision Computing 26(5), 647–656 (2008)
4. Zafeiriou, S., Tefas, A., Buciu, I., Pitas, I.: Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification. IEEE Transactions on Neural Networks 17(3), 683–695 (2006)
5. Guillamet, D., Vitria, J.: Classifying faces with nonnegative matrix factorization. In: Proc. 5th Catalan Conference for Artificial Intelligence, Castello de la Plana, Spain, pp. 24–31 (2002)
6. Benetos, E., Kotti, M., Kotropoulos, C.: Musical instrument classification using non-negative matrix factorization algorithms and subset feature selection. In: Proc. of 2006 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2006), Toulouse, France (2006)
7. Donoho, D., Stodden, V.: When does non-negative matrix factorization give a correct decomposition into parts? In: Thrun, S., Saul, L., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems (NIPS), vol. 16. MIT Press, Cambridge (2004)
8. Cai, D., He, X., Wu, X., Han, J.: Nonnegative matrix factorization on manifold. In: Proc. 8-th IEEE International Conference on Data Mining (ICDM), pp. 63–72 (2008)
9. Cai, D., He, X., Han, J., Huang, T.: Graph regularized nonnegative matrix factorization for data representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(8), 1548–1560 (2011)
10. Guan, N., Tao, D., Luo, Z., Yuan, B.: Manifold regularized discriminative nonnegative matrix factorization with fast gradient descent. IEEE Transactions on Image Processing 20(7), 2030–2048 (2011)
11. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Series in Operations Research. Springer, New York (1999)
12. 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)
13. Kim, H., Park, H.: Non-negative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal in Matrix Analysis and Applications 30(2), 713–730 (2008)
14. Birgin, E.G., Martnez, J.M., Raydan, M.: Nonmonotone spectral projected gradient methods on convex sets. SIAM Journal on Control and Optimization 10, 1196–1211 (2000)
15. Lin, C.J.: Projected gradient methods for non-negative matrix factorization. Neural Computation 19(10), 2756–2779 (2007)
16. Zdunek, R.: Spectral signal unmixing with interior-point nonnegative matrix factorization. In: Villa, A.E.P., Duch, W., Érdi, P., Masulli, F., Palm, G. (eds.) ICANN 2012, Part I. LNCS, vol. 7552, pp. 65–72. Springer, Heidelberg (2012)